

VIPA System 300S

CPU-SC | 312-5BE13 | Manual

HB140 | CPU-SC | 312-5BE13 | GB | 15-50

SPEED7 CPU 312SC

VIPA GmbH
Ohmstr. 4
91074 Herzogenaurach
Telephone: +49 9132 744-0
Fax: +49 9132 744-1864
Email: info@vipa.com
Internet: www.vipa.com

Table of contents

1	General	6
	1.1 Copyright © VIPA GmbH	6
	1.2 About this manual.....	7
	1.3 Safety information.....	8
2	Basics	10
	2.1 Safety information for users.....	10
	2.2 Operating structure of a CPU.....	10
	2.2.1 General.....	10
	2.2.2 Applications	11
	2.2.3 Operands.....	11
	2.3 CPU 312-5BE13.....	12
	2.4 General data.....	13
3	Assembly and installation guidelines	16
	3.1 Installation dimensions.....	16
	3.2 Assembly standard bus.....	17
	3.3 Cabling.....	18
	3.4 Installation guidelines.....	21
4	Hardware description	24
	4.1 Properties.....	24
	4.2 Structure.....	25
	4.2.1 General.....	25
	4.2.2 Interfaces.....	26
	4.2.3 In-/Output area CPU 312-5BE13.....	28
	4.2.4 Memory management.....	31
	4.2.5 Storage media slot	31
	4.2.6 Battery backup for clock and RAM.....	32
	4.2.7 Operating mode switch.....	32
	4.2.8 LEDs.....	33
	4.3 Technical data.....	34
5	Deployment CPU 312-5BE13	46
	5.1 Assembly.....	46
	5.2 Start-up behavior.....	46
	5.3 Addressing.....	47
	5.3.1 Overview.....	47
	5.3.2 Addressing Backplane bus I/O devices.....	47
	5.4 Address assignment.....	48
	5.5 Hardware configuration - CPU.....	49
	5.6 Hardware configuration - I/O modules.....	50
	5.7 Hardware configuration - Ethernet PG/OP channel.....	50
	5.8 CPU parametrization.....	52
	5.8.1 Parametrization via Siemens CPU.....	52
	5.8.2 CPU parameters.....	52
	5.9 Project transfer.....	54
	5.9.1 Transfer via MPI.....	55
	5.9.2 Transfer via Ethernet.....	56
	5.9.3 Transfer via MMC.....	56
	5.10 Access to the internal Web page.....	57

5.11	Operating modes.....	59
5.11.1	Overview.....	59
5.11.2	Function security.....	61
5.12	Overall reset.....	62
5.13	Firmware update.....	63
5.14	Reset to factory setting.....	66
5.15	Slot for storage media.....	67
5.16	Memory extension with MCC.....	68
5.17	Extended know-how protection.....	69
5.18	MMC-Cmd - Auto commands.....	70
5.19	VIPA specific diagnostic entries.....	72
5.20	Control and monitoring of variables with test functions..	88
6	Deployment I/O periphery.....	90
6.1	Overview.....	90
6.2	In-/Output area CPU 312-5BE13.....	91
6.3	Address assignment.....	94
6.4	Digital part.....	95
6.4.1	Access to the I/O area.....	97
6.4.2	Parameterization - Digital part.....	98
6.5	Counter.....	98
6.5.1	Counter - Fast introduction.....	98
6.5.2	SFB 47 - COUNT - Counter controlling.....	103
6.5.3	Counter - Functions.....	108
6.5.4	Counter - Additional functions.....	113
6.6	Frequency measurement.....	120
6.6.1	Overview.....	120
6.6.2	Inputs for the frequency measurement.....	121
6.6.3	Parameterization.....	122
6.6.4	SFB 48 - FREQUENC - Frequency measurement.....	123
6.7	Pulse width modulation - PWM.....	126
6.7.1	Overview.....	126
6.7.2	Parameterization.....	127
6.7.3	SFB 49 - PULSE - Pulse width modulation.....	129
6.8	Diagnostic and interrupt.....	132
6.8.1	Process interrupt.....	132
6.8.2	Diagnostic interrupt.....	134
7	Deployment PtP communication.....	141
7.1	Fast introduction.....	141
7.2	Principle of the data transfer.....	141
7.3	Deployment of RS485 interface for PtP	142
7.4	Parametrization.....	143
7.4.1	FC/SFC 216 - SER_CFG.....	143
7.5	Communication.....	146
7.5.1	Overview.....	146
7.5.2	FC/SFC 217 - SER_SND.....	147
7.5.3	FC/SFC 218 - SER_RCV.....	152
7.6	Protocols and procedures	154
7.7	Modbus - Function codes	158
7.8	Modbus - Example communication.....	163

8	WinPLC7	166
8.1	System conception.....	166
8.2	Installation.....	166
8.3	Example project engineering.....	168
8.3.1	Job definition.....	168
8.3.2	Project engineering.....	168
8.3.3	Test the PLC program in the <i>Simulator</i>	174
8.3.4	Transfer PLC program to CPU and its execution.....	175
9	Configuration with TIA Portal	177
9.1	TIA Portal - Work environment	177
9.1.1	General.....	177
9.1.2	Work environment of the TIA Portal.....	177
9.2	TIA Portal - Hardware configuration - CPU	178
9.3	TIA Portal - Hardware configuration - I/O modules.....	179
9.4	TIA Portal - Hardware configuration - Ethernet PG/OP channel.....	180
9.5	TIA Portal - Include VIPA library.....	183
9.6	TIA Portal - Project transfer.....	183

1 General

1.1 Copyright © VIPA GmbH

All Rights Reserved

This document contains proprietary information of VIPA and is not to be disclosed or used except in accordance with applicable agreements.

This material is protected by the copyright laws. It may not be reproduced, distributed, or altered in any fashion by any entity (either internal or external to VIPA), except in accordance with applicable agreements, contracts or licensing, without the express written consent of VIPA and the business management owner of the material.

For permission to reproduce or distribute, please contact: VIPA, Gesellschaft für Visualisierung und Prozessautomatisierung mbH Ohmstraße 4, D-91074 Herzogenaurach, Germany

Tel.: +49 9132 744 -0

Fax.: +49 9132 744-1864

E-Mail: info@vipa.de

<http://www.vipa.com>



Every effort has been made to ensure that the information contained in this document was complete and accurate at the time of publishing. Nevertheless, the authors retain the right to modify the information.

This customer document describes all the hardware units and functions known at the present time. Descriptions may be included for units which are not present at the customer site. The exact scope of delivery is described in the respective purchase contract.

CE Conformity Declaration

Hereby, VIPA GmbH declares that the products and systems are in compliance with the essential requirements and other relevant provisions. Conformity is indicated by the CE marking affixed to the product.

Conformity Information

For more information regarding CE marking and Declaration of Conformity (DoC), please contact your local VIPA customer service organization.

Trademarks

VIPA, SLIO, System 100V, System 200V, System 300V, System 300S, System 400V, System 500S and Commander Compact are registered trademarks of VIPA Gesellschaft für Visualisierung und Prozessautomatisierung mbH.

SPEED7 is a registered trademark of profichip GmbH.

SIMATIC, STEP, SINEC, TIA Portal, S7-300 and S7-400 are registered trademarks of Siemens AG.

Microsoft and Windows are registered trademarks of Microsoft Inc., USA.

Portable Document Format (PDF) and Postscript are registered trademarks of Adobe Systems, Inc.

All other trademarks, logos and service or product marks specified herein are owned by their respective companies.

Information product support

Contact your local VIPA Customer Service Organization representative if you wish to report errors or questions regarding the contents of this document. If you are unable to locate a customer service centre, contact VIPA as follows:

VIPA GmbH, Ohmstraße 4, 91074 Herzogenaurach, Germany

Telefax: +49 9132 744-1204

E-Mail: documentation@vipa.de

Technical support

Contact your local VIPA Customer Service Organization representative if you encounter problems with the product or have questions regarding the product. If you are unable to locate a customer service centre, contact VIPA as follows:

VIPA GmbH, Ohmstraße 4, 91074 Herzogenaurach, Germany

Tel.: +49 9132 744-1150 (Hotline)

E-Mail: support@vipa.de

1.2 About this manual**Objective and contents**

This manual describes the SPEED7 CPU-SC 312-5BE13 of the System 300S from VIPA. It contains a description of the construction, project implementation and usage.

Product	Order number	as of state:	
		CPU-HW	CPU-FW
CPU 312SC	312-5BE13	02	V3.6.0

Target audience

The manual is targeted at users who have a background in automation technology.

Structure of the manual

The manual consists of chapters. Every chapter provides a self-contained description of a specific topic.

- Guide to the document** The following guides are available in the manual:
- An overall table of contents at the beginning of the manual
 - References with page numbers
- Availability** The manual is available in:
- printed form, on paper
 - in electronic form as PDF-file (Adobe Acrobat Reader)
- Icons Headings** Important passages in the text are highlighted by following icons and headings:

**DANGER!**

Immediate or likely danger. Personal injury is possible.

**CAUTION!**

Damages to property is likely if these warnings are not heeded.



Supplementary information and useful tips.

1.3 Safety information

Applications conforming with specifications

- The system is constructed and produced for:
- communication and process control
 - industrial applications
 - operation within the environmental conditions specified in the technical data
 - installation into a cubicle

**DANGER!**

This device is not certified for applications in
– in explosive environments (EX-zone)

Documentation

- The manual must be available to all personnel in the
- project design department
 - installation department
 - commissioning
 - operation

**CAUTION!**

The following conditions must be met before using or commissioning the components described in this manual:

- Hardware modifications to the process control system should only be carried out when the system has been disconnected from power!
- Installation and hardware modifications only by properly trained personnel.
- The national rules and regulations of the respective country must be satisfied (installation, safety, EMC ...)

Disposal

National rules and regulations apply to the disposal of the unit!

2 Basics

2.1 Safety information for users

Handling of electrostatic sensitive modules

VIPA modules make use of highly integrated components in MOS-Technology. These components are extremely sensitive to over-voltages that can occur during electrostatic discharges. The following symbol is attached to modules that can be destroyed by electrostatic discharges.



The Symbol is located on the module, the module rack or on packing material and it indicates the presence of electrostatic sensitive equipment. It is possible that electrostatic sensitive equipment is destroyed by energies and voltages that are far less than the human threshold of perception. These voltages can occur where persons do not discharge themselves before handling electrostatic sensitive modules and they can damage components thereby, causing the module to become inoperable or unusable. Modules that have been damaged by electrostatic discharges can fail after a temperature change, mechanical shock or changes in the electrical load. Only the consequent implementation of protection devices and meticulous attention to the applicable rules and regulations for handling the respective equipment can prevent failures of electrostatic sensitive modules.

Shipping of modules

Modules must be shipped in the original packing material.

Measurements and alterations on electrostatic sensitive modules

When you are conducting measurements on electrostatic sensitive modules you should take the following precautions:

- Floating instruments must be discharged before use.
- Instruments must be grounded.

Modifying electrostatic sensitive modules you should only use soldering irons with grounded tips.



CAUTION!

Personnel and instruments should be grounded when working on electrostatic sensitive modules.

2.2 Operating structure of a CPU

2.2.1 General

The CPU contains a standard processor with internal program memory. In combination with the integrated SPEED7 technology the unit provides a powerful solution for process automation applications within the System 300S family. A CPU supports the following modes of operation:

- cyclic operation
- timer processing
- alarm controlled operation
- priority based processing

Cyclic processing	Cyclic processing represents the major portion of all the processes that are executed in the CPU. Identical sequences of operations are repeated in a never-ending cycle.
Timer processing	Where a process requires control signals at constant intervals you can initiate certain operations based upon a timer , e.g. not critical monitoring functions at one-second intervals.
Alarm controlled processing	If a process signal requires a quick response you would allocate this signal to an alarm controlled procedure. An alarm can activate a procedure in your program.
Priority based processing	The above processes are handled by the CPU in accordance with their priority . Since a timer or an alarm event requires a quick reaction, the CPU will interrupt the cyclic processing when these high-priority events occur to react to the event. Cyclic processing will resume, once the reaction has been processed. This means that cyclic processing has the lowest priority.

2.2.2 Applications

The program that is present in every CPU is divided as follows:

- System routine
- User application

System routine

The system routine organizes all those functions and procedures of the CPU that are not related to a specific control application.

User application

This consists of all the functions that are required for the processing of a specific control application. The operating modules provide the interfaces to the system routines.

2.2.3 Operands

The following series of operands is available for programming the CPU:

- Process image and periphery
- Bit memory
- Timers and counters
- Data blocks

Process image and periphery

The user application can quickly access the process image of the inputs and outputs PIO/PII. You may manipulate the following types of data:

- individual Bits
- Bytes
- Words
- Double words

You may also gain direct access to peripheral modules via the bus from user application. The following types of data are available:

- Bytes
- Words
- Blocks

Bit Memory

The bit memory is an area of memory that is accessible by means of certain operations. Bit memory is intended to store frequently used working data.

You may access the following types of data:

- individual Bits
- Bytes
- Words
- Double words

Timers and counters

In your program you may load cells of the timer with a value between 10ms and 9990s. As soon as the user application executes a start-operation, the value of this timer is decremented by the interval that you have specified until it reaches zero.

You may load counter cells with an initial value (max. 999) and increment or decrement these when required.

Data Blocks

A data block contains constants or variables in the form of bytes, words or double words. You may always access the current data block by means of operands.

You may access the following types of data:

- individual Bits
- Bytes
- Words
- Double words

2.3 CPU 312-5BE13

Overview

The CPU 312-5BE13 bases upon the SPEED7 technology. This supports the CPU at programming and communication by means of co-processors that causes a power improvement for highest needs.

- The CPU is programmed in STEP[®]7 from Siemens. For this you may use the SIMATIC Manager or TIA Portal from Siemens. Here the instruction set of the S7-400 from Siemens is used.
- Modules and CPUs of the System 300S from VIPA and Siemens may be used at the bus as a mixed configuration.
- The user application is stored in the battery buffered RAM or on an additionally pluggable MMC storage module.
- The CPU 312-5BE13 is configured as CPU 312C (6ES7 312-5BE13-0AB0/V2.6) from Siemens.

Memory	<p>The CPU has an integrated memory. Information about the capacity of the memory may be found at the front of the CPU. The memory is divided into the following parts:</p> <ul style="list-style-type: none"> ■ Load memory 512kbyte ■ Code memory (50% of the work memory) ■ Data memory (50% of the work memory) ■ Work memory 64kbyte <ul style="list-style-type: none"> – There is the possibility to extend the work memory to its maximum printed capacity 512kbyte by means of a MCC memory extension card.
Integrated PtP function	<p>The CPU has a RS485 interface, which is fix set to PtP communication (point to point). This supports the serial process connection to different source or destination systems.</p>
Integrated Ethernet PG/OP channel	<p>The CPU has an Ethernet interface for PG/OP communication. After assigning IP address parameters with your configuration tool, via the "PLC" functions you may directly access the Ethernet PG/OP channel and program res. remote control your CPU. You may also access the CPU with a visualization software via these connections.</p>
Operation Security	<ul style="list-style-type: none"> ■ Wiring by means of spring pressure connections (CageClamps) at the front connector ■ Core cross-section 0.08...2.5mm² ■ Total isolation of the wiring at module change ■ Potential separation of all modules to the backplane bus
Dimensions/ Weight	<p>Dimensions of the basic enclosure:</p> <ul style="list-style-type: none"> ■ 2tier width: (WxHxD) in mm: 80x125x120
Integrated power supply	<p>The CPU comes with an integrated power supply. The power supply is to be supplied with DC 24V. By means of the supply voltage, the internal electronic is supplied as well as the connected modules via backplane bus. The power supply is protected against inverse polarity and overcurrent.</p>

2.4 General data

Conformity and approval		
Conformity		
CE	2006/95/EG	Low-voltage directive
	2004/108/EG	EMC directive
Approval		
UL	UL 508	Approval for USA and Canada

General data

Conformity and approval

others		
RoHS	2011/65/EU	Product is lead-free; Restriction of the use of certain hazardous substances in electrical and electronic equipment

Protection of persons and device protection

Type of protection	-	IP20
Electrical isolation		
to the field bus	-	electrically isolated
to the process level	-	electrically isolated
Insulation resistance		-
Insulation voltage to reference earth		
Inputs / outputs	-	AC / DC 50V, test voltage AC 500V
Protective measures	-	against short circuit

Environmental conditions to EN 61131-2

Climatic		
Storage / transport	EN 60068-2-14	-25...+70°C
Operation		
Horizontal installation hanging	EN 61131-2	0...+60°C
Horizontal installation lying	EN 61131-2	0...+55°C
Vertical installation	EN 61131-2	0...+50°C
Air humidity	EN 60068-2-30	RH1 (without condensation, rel. humidity 10... 95%)
Pollution	EN 61131-2	Degree of pollution 2
Installation altitude max.	-	2000m
Mechanical		
Oscillation	EN 60068-2-6	1g, 9Hz ... 150Hz
Shock	EN 60068-2-27	15g, 11ms

Mounting conditions

Mounting place	-	In the control cabinet
Mounting position	-	Horizontal and vertical

EMC	Standard	Comment
Emitted interference	EN 61000-6-4	Class A (Industrial area)
Noise immunity zone B	EN 61000-6-2	Industrial area
	EN 61000-4-2	ESD 8kV at air discharge (degree of severity 3), 4kV at contact discharge (degree of severity 2)
	EN 61000-4-3	HF field immunity (casing) 80MHz ... 1000MHz, 10V/m, 80% AM (1kHz) 1.4GHz ... 2.0GHz, 3V/m, 80% AM (1kHz) 2GHz ... 2.7GHz, 1V/m, 80% AM (1kHz)
	EN 61000-4-6	HF conducted 150kHz ... 80MHz, 10V, 80% AM (1kHz)
	EN 61000-4-4	Burst, degree of severity 3
	EN 61000-4-5	Surge, installation class 3 *

*) Due to the high-energetic single pulses with Surge an appropriate external protective circuit with lightning protection elements like conductors for lightning and overvoltage is necessary.

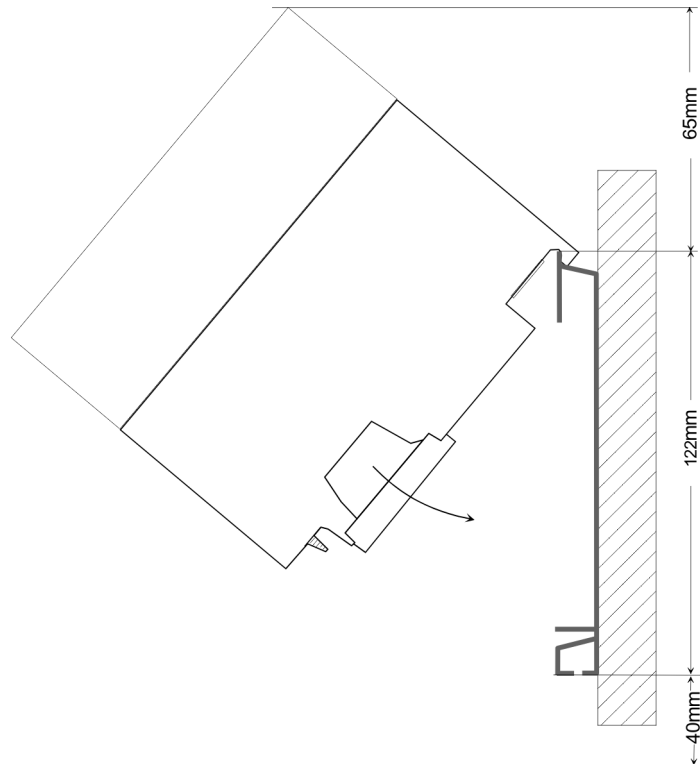
3 Assembly and installation guidelines

3.1 Installation dimensions

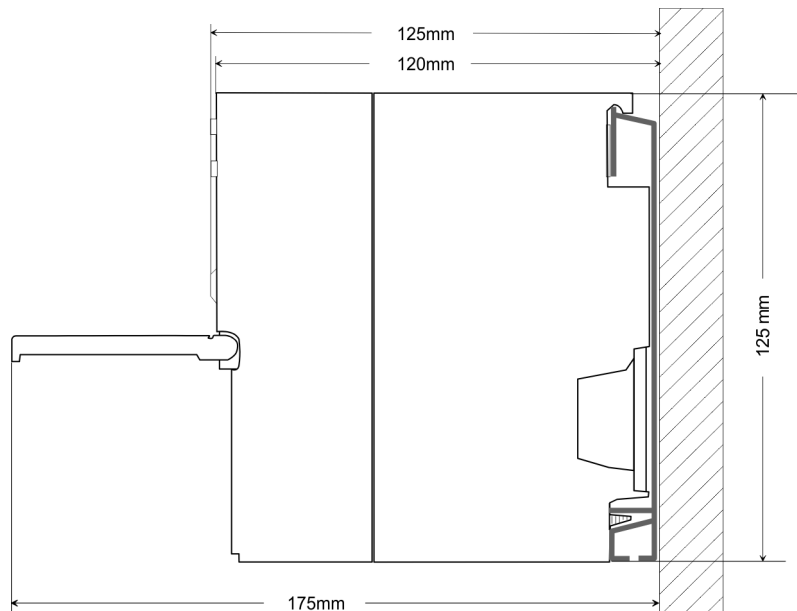
Dimensions Basic enclosure

2tier width (WxHxD) in mm: 80 x 125 x 120

Dimensions



Installation dimensions



3.2 Assembly standard bus

General

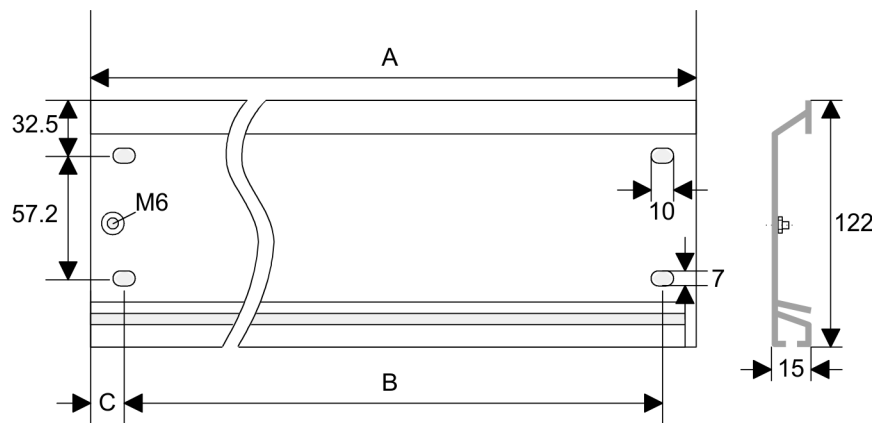
The single modules are directly installed on a profile rail and connected via the backplane bus connector. Before installing the modules you have to clip the backplane bus connector to the module from the backside. The backplane bus connector is delivered together with the peripheral modules.

Profile rail

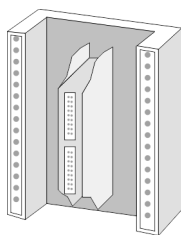
Order number	A	B	C
390-1AB60	160	140	10
390-1AE80	482	466	8.3
390-1AF30	530	500	15
390-1AJ30	830	800	15
390-9BC00*	2000	Drillings only left	15

*) Unit pack: 10 pieces

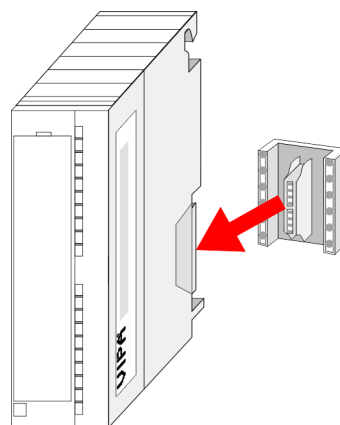
Measures in mm



Bus connector

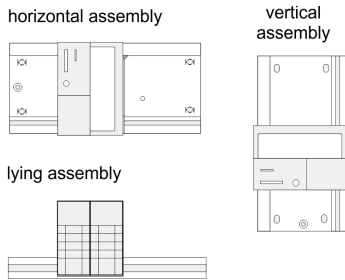


For the communication between the modules the System 300S uses a backplane bus connector. Backplane bus connectors are included in the delivering of the peripheral modules and are clipped at the module from the backside before installing it to the profile rail.



Cabling

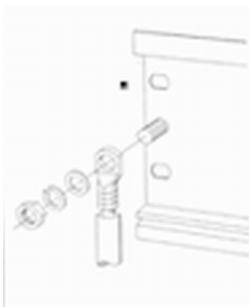
Assembly possibilities



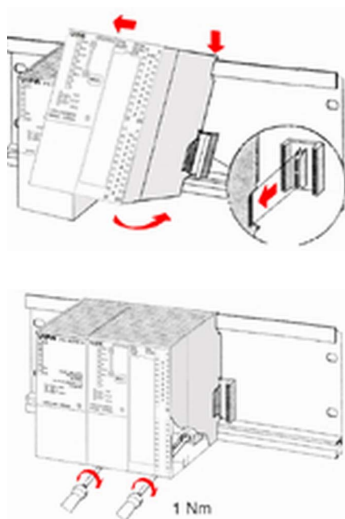
Please regard the allowed environment temperatures:

- horizontal assembly: from 0 to 60°C
- vertical assembly: from 0 to 50°C
- lying assembly: from 0 to 55°C

Approach



1. Bolt the profile rail with the background (screw size: M6), so that you still have minimum 65mm space above and 40mm below the profile rail.
2. If the background is a grounded metal or device plate, please look for a low-impedance connection between profile rail and background.
3. Connect the profile rail with the protected earth conductor. For this purpose there is a bolt with M6-thread.
4. The minimum cross-section of the cable to the protected earth conductor has to be 10mm².
5. Stick the power supply to the profile rail and pull it to the left side to the grounding bolt of the profile rail.
6. Fix the power supply by screwing.
7. Take a backplane bus connector and click it at the CPU from the backside like shown in the picture.
8. Stick the CPU to the profile rail right from the power supply and pull it to the power supply.
9. Click the CPU downwards and bolt it like shown.
10. Repeat this procedure with the peripheral modules, by clicking a backplane bus connector, stick the module right from the modules you've already fixed, click it downwards and connect it with the backplane bus connector of the last module and bolt it.



3.3 Cabling



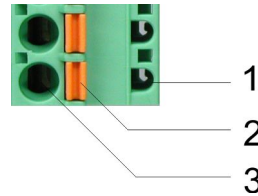
CAUTION!

- The power supplies must be released before installation and repair tasks, i.e. before handling with the power supply or with the cabling you must disconnect current/voltage (pull plug, at fixed connection switch off the concerning fuse)!
- Installation and modifications only by properly trained personnel!

CageClamp technology (green)

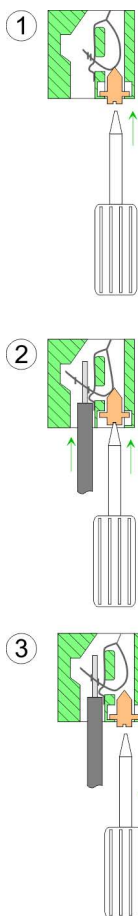
For the cabling of power supply of a CPU, a green plug with Cage-Clamp technology is deployed. The connection clamp is realized as plug that may be clipped off carefully if it is still cabled.

Here wires with a cross-section of 0.08mm² to 2.5mm² may be connected. You can use flexible wires without end case as well as stiff wires.



- 1 Test point for 2mm test tip
- 2 Locking (orange) for screwdriver
- 3 Round opening for wires

The picture on the left side shows the cabling step by step from top view.



- 1. ➤ For cabling you push the locking vertical to the inside with a suitable screwdriver and hold the screwdriver in this position.
- 2. ➤ Insert the de-isolated wire into the round opening. You may use wires with a cross-section from 0.08mm² to 2.5mm²
- 3. ➤ By removing the screwdriver the wire is connected safely with the plug connector via a spring.

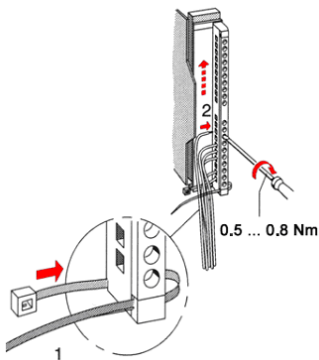
Front connectors of the in-/output modules

In the following the cabling of the two variants are shown.

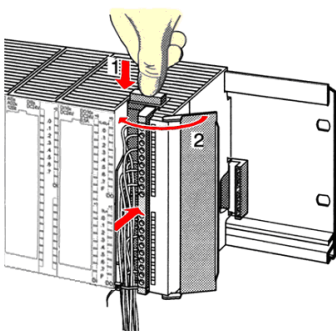
20pole screw connection 392-1AJ00



1. ▶ Open the front flap of your I/O module.
2. ▶ Bring the front connector in cabling position.
For this you plug the front connector on the module until it locks. In this position the front connector juts out of the module and has no contact yet.
3. ▶ De-isolate your wires. If needed, use core end cases.
4. ▶ Thread the included cable binder into the front connector.
5. ▶ If you want to lead out your cables from the bottom of the module, start with the cabling from bottom to top, res. from top to bottom, if the cables should be led out at the top.
6. ▶ Bolt also the connection screws of not cabled screw clamps.



7. ▶ Fix the cable binder for the cable bundle.

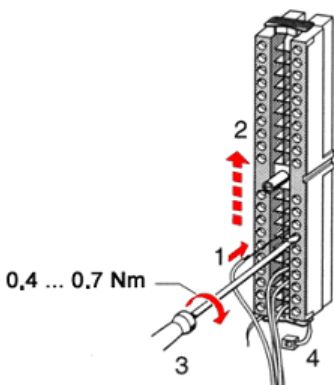


8. ▶ Push the release key at the front connector on the upper side of the module and at the same time push the front connector into the module until it locks.
9. ▶ Now the front connector is electrically connected with your module.
10. ▶ Close the front flap.
11. ▶ Fill out the labeling strip to mark the single channels and push the strip into the front flap.

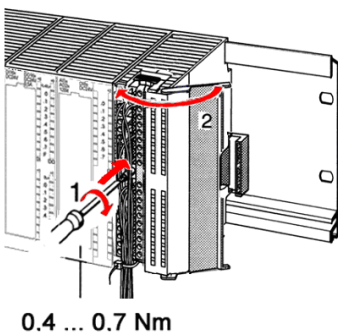
40pole screw connection 392-1AM00



1. ▶ Open the front flap of your I/O module.
2. ▶ Bring the front connector in cabling position.
For this you plug the front connector on the module until it locks. In this position the front connector juts out of the module and has no contact yet.
3. ▶ De-isolate your wires. If needed, use core end cases.
4. ▶ If you want to lead out your cables from the bottom of the module, start with the cabling from bottom to top, res. from top to bottom, if the cables should be led out at the top.
5. ▶ Bolt also the connection screws of not cabled screw clamps.



6. ▶ Put the included cable binder around the cable bundle and the front connector.
7. ▶ Fix the cable binder for the cable bundle.



8. ▶ Bolt the fixing screw of the front connector.
9. ▶ Now the front connector is electrically connected with your module.
10. ▶ Close the front flap.
11. ▶ Fill out the labeling strip to mark the single channels and push the strip into the front flap.

3.4 Installation guidelines

General

The installation guidelines contain information about the interference free deployment of a PLC system. There is the description of the ways, interference may occur in your PLC, how you can make sure the electromagnetic compatibility (EMC), and how you manage the isolation.

What does EMC mean?	<p>Electromagnetic compatibility (EMC) means the ability of an electrical device, to function error free in an electromagnetic environment without being interfered respectively without interfering the environment.</p> <p>The components of VIPA are developed for the deployment in industrial environments and meets high demands on the EMC. Nevertheless you should project an EMC planning before installing the components and take conceivable interference causes into account.</p>
Possible interference causes	<p>Electromagnetic interferences may interfere your control via different ways:</p> <ul style="list-style-type: none"> ■ Electromagnetic fields (RF coupling) ■ Magnetic fields with power frequency ■ Bus system ■ Power supply ■ Protected earth conductor <p>Depending on the spreading medium (lead bound or lead free) and the distance to the interference cause, interferences to your control occur by means of different coupling mechanisms.</p> <p>There are:</p> <ul style="list-style-type: none"> ■ galvanic coupling ■ capacitive coupling ■ inductive coupling ■ radiant coupling
Basic rules for EMC	<p>In the most times it is enough to take care of some elementary rules to guarantee the EMC. Please regard the following basic rules when installing your PLC.</p> <ul style="list-style-type: none"> ■ Take care of a correct area-wide grounding of the inactive metal parts when installing your components. <ul style="list-style-type: none"> – Install a central connection between the ground and the protected earth conductor system. – Connect all inactive metal extensive and impedance-low. – Please try not to use aluminium parts. Aluminium is easily oxidizing and is therefore less suitable for grounding. ■ When cabling, take care of the correct line routing. <ul style="list-style-type: none"> – Organize your cabling in line groups (high voltage, current supply, signal and data lines). – Always lay your high voltage lines and signal respectively data lines in separate channels or bundles. – Route the signal and data lines as near as possible beside ground areas (e.g. suspension bars, metal rails, tin cabinet). ■ Proof the correct fixing of the lead isolation. <ul style="list-style-type: none"> – Data lines must be laid isolated. – Analog lines must be laid isolated. When transmitting signals with small amplitudes the one sided laying of the isolation may be favourable. – Lay the line isolation extensively on an isolation/protected earth conductor rail directly after the cabinet entry and fix the isolation with cable clamps. – Make sure that the isolation/protected earth conductor rail is connected impedance-low with the cabinet. – Use metallic or metallised plug cases for isolated data lines.

- In special use cases you should appoint special EMC actions.
 - Consider to wire all inductivities with erase links.
 - Please consider luminescent lamps can influence signal lines.
- Create a homogeneous reference potential and ground all electrical operating supplies when possible.
 - Please take care for the targeted employment of the grounding actions. The grounding of the PLC serves for protection and functionality activity.
 - Connect installation parts and cabinets with your PLC in star topology with the isolation/protected earth conductor system. So you avoid ground loops.
 - If there are potential differences between installation parts and cabinets, lay sufficiently dimensioned potential compensation lines.

Isolation of conductors

Electrical, magnetically and electromagnetic interference fields are weakened by means of an isolation, one talks of absorption. Via the isolation rail, that is connected conductive with the rack, interference currents are shunt via cable isolation to the ground. Here you have to make sure, that the connection to the protected earth conductor is impedance-low, because otherwise the interference currents may appear as interference cause.

When isolating cables you have to regard the following:

- If possible, use only cables with isolation tangle.
- The hiding power of the isolation should be higher than 80%.
- Normally you should always lay the isolation of cables on both sides. Only by means of the both-sided connection of the isolation you achieve high quality interference suppression in the higher frequency area. Only as exception you may also lay the isolation one-sided. Then you only achieve the absorption of the lower frequencies. A one-sided isolation connection may be convenient, if:
 - the conduction of a potential compensating line is not possible.
 - analog signals (some mV respectively μA) are transferred.
 - foil isolations (static isolations) are used.
- With data lines always use metallic or metallised plugs for serial couplings. Fix the isolation of the data line at the plug rack. Do not lay the isolation on the PIN 1 of the plug bar!
- At stationary operation it is convenient to strip the insulated cable interruption free and lay it on the isolation/protected earth conductor line.
- To fix the isolation tangles use cable clamps out of metal. The clamps must clasp the isolation extensively and have well contact.
- Lay the isolation on an isolation rail directly after the entry of the cable in the cabinet. Lead the isolation further on to your PLC and don't lay it on there again!



CAUTION!

Please regard at installation!

At potential differences between the grounding points, there may be a compensation current via the isolation connected at both sides.

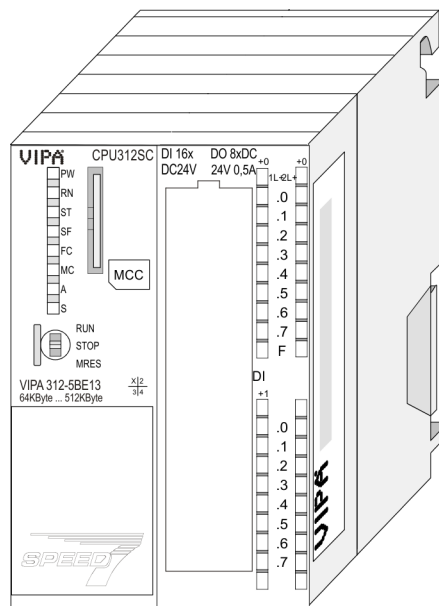
Remedy: Potential compensation line

4 Hardware description

4.1 Properties

CPU 312-5BE13

- SPEED7 technology integrated
- 64kbyte work memory integrated (32kbyte code, 32kbyte data)
- Work memory expandable to max. 512Mbyte (256kbyte code, 256kbyte data)
- 512kbyte load memory
- RS485 interface for PtP communication
- Ethernet PG/OP interface integrated
- MPI interface
- MCC slot for external memory cards (lockable)
- Status LEDs for operating state and diagnostics
- Real-time clock battery buffered
- Digital I/Os: DI 16xDC24V / DO 8xDC 24V, 0.5A
- 2 channels for counter, frequency measurement and pulse width modulation
- I/O address range digital/analog 1024byte
- 512 timer
- 512 counter
- 8192 flag byte



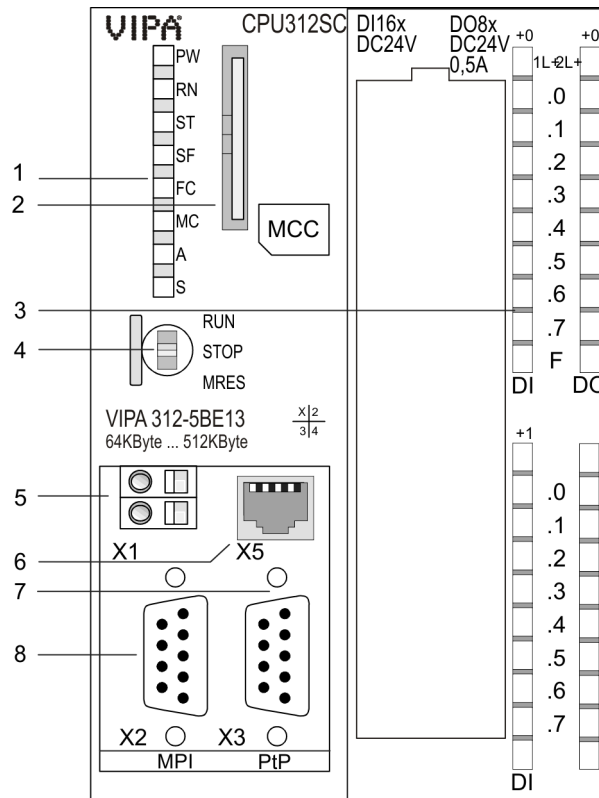
Ordering data

Type	Order number	Description
312SC	312-5BE13	MPI interface, card slot, real time clock, Ethernet interface for PG/OP, DI 16xDC24V / DO 8xDC24V, 0.5A, 2 channels technological functions

4.2 Structure

4.2.1 General

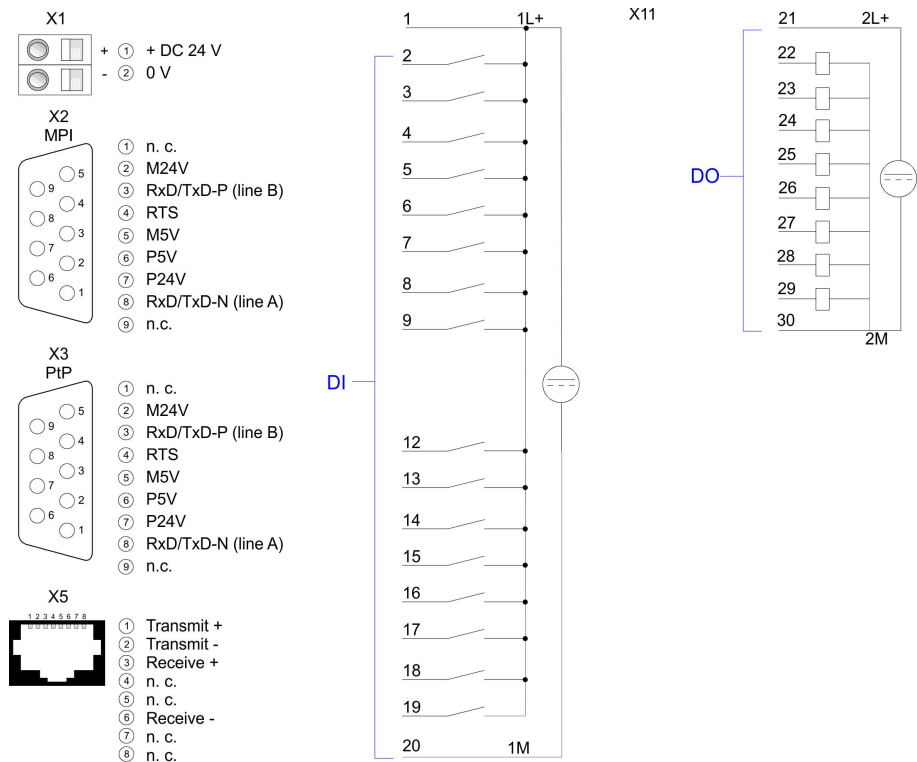
CPU 312-5BE13



- 1 LEDs of the CPU part
- 2 Storage media slot (lockable)
- 3 LEDs of the I/O part
- 4 Operating mode switch CPU
- 5 Slot for DC 24V power supply
- 6 Twisted pair interface for Ethernet PG/OP channel
- 7 PtP interface
- 8 MPI interface

The components 5 - 8 are under the front flap!

4.2.2 Interfaces



X1: Power supply

The CPU has an integrated power supply:

- The power supply has to be provided with DC 24V. For this serves the double DC 24V slot, that is underneath the flap.
- Via the power supply not only the internal electronic is provided with voltage, but by means of the backplane bus also the connected modules.
- The power supply is protected against polarity inversion and over-current.
- The internal electronic is galvanically connected with the supply voltage.

X2: MPI interface

9pin SubD jack:

- The MPI interface serves for the connection between programming unit and CPU.
- By means of this the project engineering and programming happens.
- MPI serves for communication between several CPUs or between HMIs and CPU.
- Standard setting is MPI Address 2.

X3: PtP interface

9pin SubD jack:

The CPU has a PtP interface. The interface is fix set to PtP communication.

- PtP functionality
 - Using the PtP functionality the RS485 interface is allowed to connect via serial point-to-point connection to different source res. target systems.
 - Here the following protocols are supported: ASCII, STX/ETX, 3964R, USS and Modbus-Master (ASCII, RTU).
 - The PtP communication is configured during run-time by means of the SFC 216 (SER_CFG). The communication happens by means of the SFC 217 (SER_SND) and SFC 218 (SER_RCV).

X5: Ethernet PG/OP channel

8pin RJ45 jack:

- The RJ45 jack serves the interface to the Ethernet PG/OP channel.
- This interface allows you to program res. remote control your CPU, to access the internal web site or to connect a visualization.
- Configurable connections are not possible.
- For online access to the CPU via Ethernet PG/OP channel valid IP address parameters have to be assigned to this.

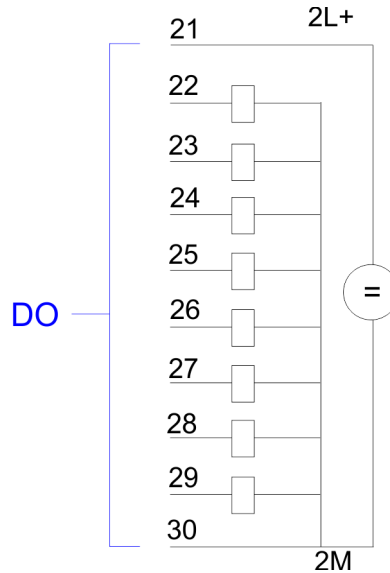
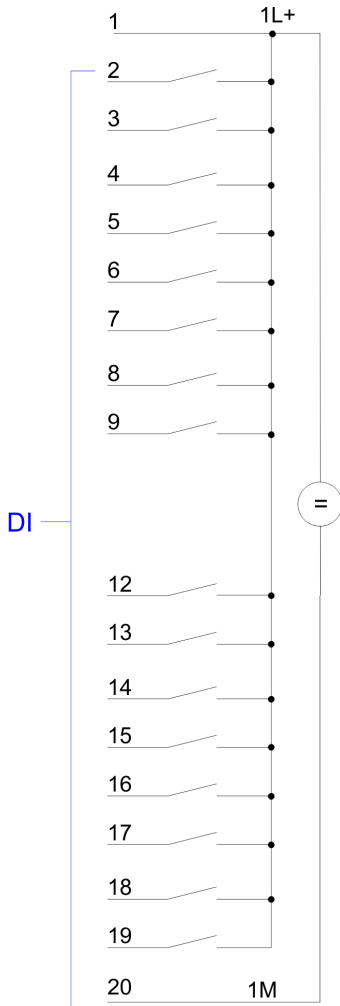
4.2.3 In-/Output area CPU 312-5BE13


Overview

The CPU 312-5BE13 has the following digital in and output channels integrated in one casing:

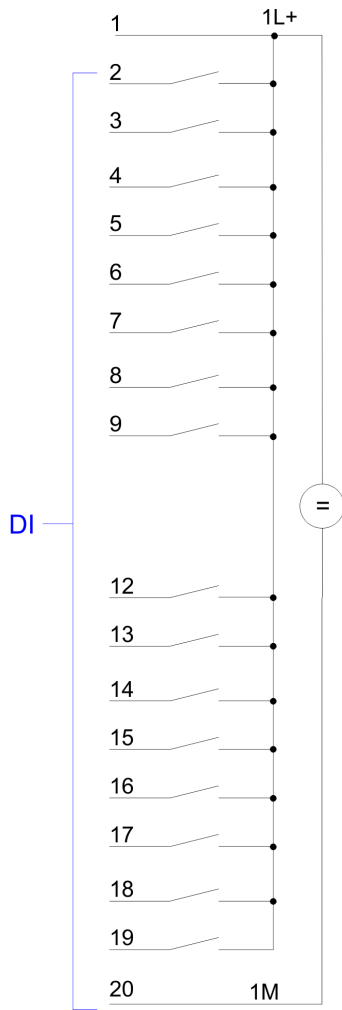
- Digital Input: 16xDC 24V, with interrupt capability
- Digital Output: 8xDC 24V, 0.5A
- Technological functions: 2 channels

X11:



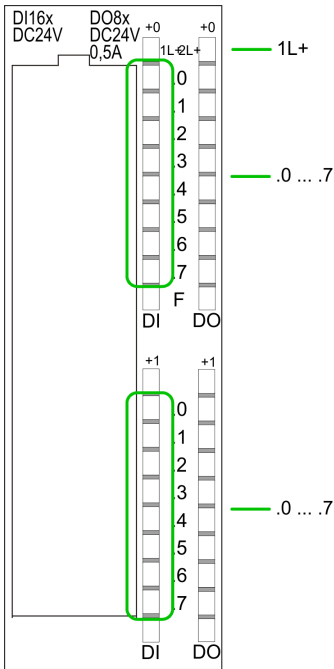


CAUTION! Please regard that the voltage at an output channel is always \leq the supply voltage connected to L+.



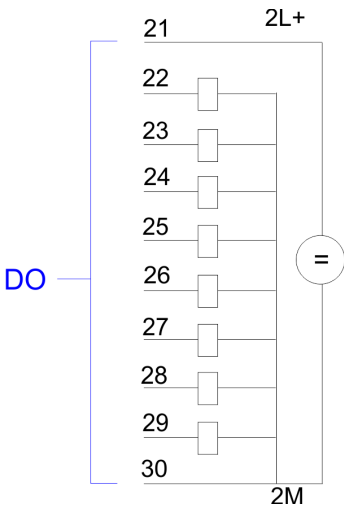
Pin assignment X11: DI

Pin	Assignment
1	1L+ Power supply +DC 24V
2	I+0.0 / Channel 0 (A) / Pulse
3	I+0.1 / Channel 0 (B) / Direction
4	I+0.2 / Channel 0 HW gate
5	I+0.3 / Channel 1 (A) / Pulse
6	I+0.4 / Channel 1 (B) / Direction
7	I+0.5 / Channel 1 HW gate
8	I+0.6
9	I+0.7
10	not used
11	not used
12	I+1.0
13	I+1.1
14	I+1.2
15	I+1.3
16	I+1.4 / Channel 0 Latch
17	I+1.5 / Channel 1 Latch
18	I+1.6
19	I+1.7
20	Ground 1M DI



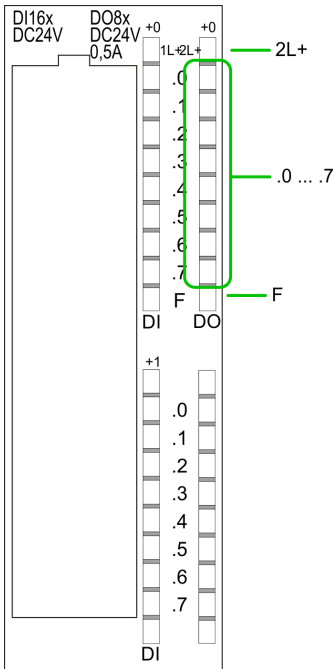
Status indication X11: DI

- 1L+
 - LED (green)
 - Supply voltage available for DI
- .07
 - LEDs (green)
 - I+0.0 ... I+0.7
 - I+1.0 ... I+1.7
 - Starting with ca. 15V the signal "1" at the input is recognized and the according LED is activated



Pin assignment X11: DO

Pin	Assignment
21	2L+ Power supply +DC 24V
22	O+0.0 / Channel 0 Output
23	O+0.1 / Channel 1 Output
24	Q+0.2
25	Q+0.3
26	Q+0.4
27	Q+0.5
28	Q+0.6
29	Q+0.7
30	Ground 2M DO
31 ... 40	not used



Status indication X11: DO

- 2L+
 - LED (green)
 - Supply voltage available for DO
- .07
 - LEDs (green)
 - Q+0.0 ... Q+0.7
 - The according LED is on at active output
- F
 - LED (red)
 - Overload or short circuit error

4.2.4 Memory management

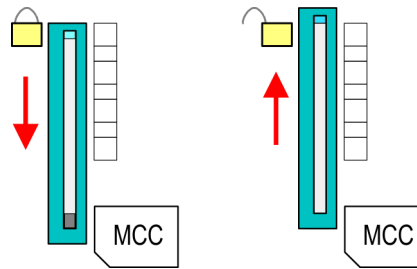
Memory

The CPU has an integrated memory. Information about the capacity of the memory may be found at the front of the CPU. The memory is divided into the following parts:

- Load memory 512kbyte
- Code memory (50% of the work memory)
- Data memory (50% of the work memory)
- Work memory 64kbyte
 - There is the possibility to extend the work memory to its maximum printed capacity 512kbyte by means of a MCC memory extension card.

4.2.5 Storage media slot

- As external storage medium for applications and firmware you may use a MMC storage module (**Multimedia card**).
- The VIPA storage media are pre-formatted with the PC format FAT16 and can be accessed via a card reader.
- After PowerON respectively an overall reset the CPU checks, if there is a storage medium with data valid for the CPU.
- Push the memory card into the slot until it snaps in leaded by a spring mechanism. This ensures contacting.
- By sliding down the sliding mechanism, a just installed memory card can be protected against drop out.
- To remove, slide the sliding mechanism up again and push the storage media against the spring pressure until it is unlocked with a click.



CAUTION!
 If the media was already unlocked by the spring mechanism, with shifting the sliding mechanism, a just installed memory card can jump out of the slot!

4.2.6 Battery backup for clock and RAM

A rechargeable battery is installed on every CPU to safeguard the contents of the RAM when power is removed. This battery is also used to buffer the internal clock. The rechargeable battery is maintained by a charging circuit that receives its power from the internal power supply and that maintain the clock and RAM for a max. period of 30 days.

i

- Please connect the CPU at least for 24 hours to the power supply, so that the internal accumulator/battery is loaded accordingly.
- Please note that in case of repeated discharge cycles (charging/buffering) can reduce the buffer time continuously. Only after a charging time of 24 hours there is a buffer for max. 30 days.

CAUTION!

- After a power reset and with an empty battery the CPU starts with a BAT error and executes an overall reset. The loading procedure is not influenced by the BAT error.
- The BAT error can be deleted again, if once during power cycle the time between switching on and off the power supply is at least 30sec. and the battery is fully loaded. Otherwise with a short power cycle the BAT error still exists and an overall reset is executed.

4.2.7 Operating mode switch








- With the operating mode switch you may switch the CPU between STOP and RUN.
- During the transition from STOP to RUN the operating mode START-UP is driven by the CPU.
- Placing the switch to MRES (Memory Reset), you request an overall reset with following load from MMC, if a project there exists.

4.2.8 LEDs

LEDs CPU

As soon as the CPU is supplied with 5V, the green PW-LED (Power) is on.

RN (RUN)	ST (STOP)	SF (SFAIL)	FC (FRCE)	MC (MMC)	Meaning
green 	yellow 	red 	yellow 	yellow 	
Boot-up after PowerON					
●	BB*	●	●	●	* Blinking with 10Hz: Firmware is loaded.
●	●	●	●	●	Initialization: Phase 1
●	●	●	●	○	Initialization: Phase 2
●	●	●	○	○	Initialization: Phase 3
○	●	●	○	○	Initialization: Phase 4
Operation					
○	●	X	X	X	CPU is in STOP state.
BB	○	X	X	X	CPU is in start-up state, the RUN LED blinks during operating OB100 at least for 3s.
●	○	○	X	X	CPU is in state RUN without error.
X	X	●	X	X	There is a system fault. More information may be found in the diagnostics buffer of the CPU.
X	X	X	●	X	Variables are forced.
X	X	X	X	●	Access to the memory card.
X	BB*	○	○	○	* Blinking with 10Hz: Configuration is loaded.
Overall reset					
○	BB	X	X	X	Overall reset is requested.
○	BB*	X	X	X	* Blinking with 5Hz: Overall reset is executed.
Factory reset					
●	●	○	○	○	Factory reset is executed.
○	●	●	●	●	Factory reset finished without error.
Firmware update					
○	●	BB	BB	●	The alternate blinking indicates that there is new firmware on the memory card.
○	○	BB	BB	●	The alternate blinking indicates that a firmware update is executed.
○	●	●	●	●	Firmware update finished without error.

Technical data

RN (RUN)	ST (STOP)	SF (SFAIL)	FC (FRCE)	MC (MMC)	Meaning
○	BB*	BB*	BB*	BB*	* Blinking with 10Hz: Error during Firmware update.

on: ● | off: ○ | blinking (2Hz): BB | not relevant: X

LEDs Ethernet PG/OP channel L/A, S

The green L/A-LED (Link/Activity) indicates the physical connection of the Ethernet PG/OP channel to Ethernet. Irregular flashing of the L/A-LED indicates communication of the Ethernet PG/OP channel via Ethernet.

If the green S-LED (Speed) is on, the Ethernet PG/OP has a communication speed of 100MBit/s otherwise 10MBit/s.

4.3 Technical data



Please consider with the configuration with the Siemens TIA Portal the number of timer and counters is limited to the maximum possible number of the corresponding Siemens CPU.

Order no.	312-5BE13
Type	CPU 312SC
SPEED-Bus	-
Technical data power supply	
Power supply (rated value)	DC 24 V
Power supply (permitted range)	DC 20.4...28.8 V
Reverse polarity protection	✓
Current consumption (no-load operation)	135 mA
Current consumption (rated value)	500 mA
Inrush current	11 A
I ² t	0.7 A ² s
Max. current drain at backplane bus	3 A
Power loss	8 W
Technical data digital inputs	
Number of inputs	16
Cable length, shielded	1000 m
Cable length, unshielded	600 m
Rated load voltage	DC 24 V
Reverse polarity protection of rated load voltage	✓

Order no.	312-5BE13
Current consumption from load voltage L+ (without load)	70 mA
Rated value	DC 24 V
Input voltage for signal "0"	DC 0...5 V
Input voltage for signal "1"	DC 15...28.8 V
Input voltage hysteresis	-
Frequency range	-
Input resistance	-
Input current for signal "1"	6 mA
Connection of Two-Wire-BEROs possible	✓
Max. permissible BERO quiescent current	1.5 mA
Input delay of "0" to "1"	0.1 / 0.35 ms
Input delay of "1" to "0"	0.1 / 0.35 ms
Number of simultaneously utilizable inputs horizontal configuration	16
Number of simultaneously utilizable inputs vertical configuration	16
Input characteristic curve	IEC 61131-2, type 1
Initial data size	2 Byte
Technical data digital outputs	
Number of outputs	8
Cable length, shielded	1000 m
Cable length, unshielded	600 m
Rated load voltage	DC 24 V
Reverse polarity protection of rated load voltage	-
Current consumption from load voltage L+ (without load)	100 mA
Total current per group, horizontal configuration, 40°C	3 A
Total current per group, horizontal configuration, 60°C	2 A
Total current per group, vertical configuration	2 A
Output voltage signal "1" at min. current	L+ (-0.8 V)
Output voltage signal "1" at max. current	L+ (-0.8 V)
Output current at signal "1", rated value	0.5 A
Output current, permitted range to 40°C	5 mA to 0.6 A
Output current, permitted range to 60°C	5 mA to 0.6 A

Technical data

Order no.	312-5BE13
Output current at signal "0" max. (residual current)	0.5 mA
Output delay of "0" to "1"	100 µs
Output delay of "1" to "0"	100 µs
Minimum load current	-
Lamp load	5 W
Parallel switching of outputs for redundant control of a load	possible
Parallel switching of outputs for increased power	not possible
Actuation of digital input	✓
Switching frequency with resistive load	max. 2.5 kHz
Switching frequency with inductive load	max. 0.5 Hz
Switching frequency on lamp load	max. 2.5 kHz
Internal limitation of inductive shut-off voltage	L+ (-52 V)
Short-circuit protection of output	yes, electronic
Trigger level	1 A
Number of operating cycle of relay outputs	-
Switching capacity of contacts	-
Output data size	1 Byte
Technical data analog inputs	
Number of inputs	-
Cable length, shielded	-
Rated load voltage	-
Reverse polarity protection of rated load voltage	-
Current consumption from load voltage L+ (without load)	-
Voltage inputs	-
Min. input resistance (voltage range)	-
Input voltage ranges	-
Operational limit of voltage ranges	-
Operational limit of voltage ranges with SFU	-
Basic error limit voltage ranges	-
Basic error limit voltage ranges with SFU	-
Destruction limit current	-
Current inputs	-
Max. input resistance (current range)	-

Order no.	312-5BE13
Input current ranges	-
Operational limit of current ranges	-
Operational limit of current ranges with SFU	-
Basic error limit current ranges	-
Radical error limit current ranges with SFU	-
Destruction limit current inputs (electrical current)	-
Destruction limit current inputs (voltage)	-
Resistance inputs	-
Resistance ranges	-
Operational limit of resistor ranges	-
Operational limit of resistor ranges with SFU	-
Basic error limit	-
Basic error limit with SFU	-
Destruction limit resistance inputs	-
Resistance thermometer inputs	-
Resistance thermometer ranges	-
Operational limit of resistance thermometer ranges	-
Operational limit of resistance thermometer ranges with SFU	-
Basic error limit thermoresistor ranges	-
Basic error limit thermoresistor ranges with SFU	-
Destruction limit resistance thermometer inputs	-
Thermocouple inputs	-
Thermocouple ranges	-
Operational limit of thermocouple ranges	-
Operational limit of thermocouple ranges with SFU	-
Basic error limit thermoelement ranges	-
Basic error limit thermoelement ranges with SFU	-
Destruction limit thermocouple inputs	-
Programmable temperature compensation	-
External temperature compensation	-
Internal temperature compensation	-
Technical unit of temperature measurement	-

Technical data

Order no.	312-5BE13
Resolution in bit	-
Measurement principle	-
Basic conversion time	-
Noise suppression for frequency	-
Initial data size	-
Technical data analog outputs	
Number of outputs	-
Cable length, shielded	-
Rated load voltage	-
Reverse polarity protection of rated load voltage	-
Current consumption from load voltage L+ (without load)	-
Voltage output short-circuit protection	-
Voltage outputs	-
Min. load resistance (voltage range)	-
Max. capacitive load (current range)	-
Max. inductive load (current range)	-
Output voltage ranges	-
Operational limit of voltage ranges	-
Basic error limit voltage ranges with SFU	-
Destruction limit against external applied voltage	-
Current outputs	-
Max. in load resistance (current range)	-
Max. inductive load (current range)	-
Typ. open circuit voltage current output	-
Output current ranges	-
Operational limit of current ranges	-
Radical error limit current ranges with SFU	-
Destruction limit against external applied voltage	-
Settling time for ohmic load	-
Settling time for capacitive load	-
Settling time for inductive load	-
Resolution in bit	-
Conversion time	-
Substitute value can be applied	-

Order no.	312-5BE13
Output data size	-
Technical data counters	
Number of counters	2
Counter width	32 Bit
Maximum input frequency	10 kHz
Maximum count frequency	10 kHz
Mode incremental encoder	✓
Mode pulse / direction	✓
Mode pulse	✓
Mode frequency counter	✓
Mode period measurement	✓
Gate input available	✓
Latch input available	✓
Reset input available	-
Counter output available	✓
Load and working memory	
Load memory, integrated	512 KB
Load memory, maximum	512 KB
Work memory, integrated	64 KB
Work memory, maximal	512 KB
Memory divided in 50% program / 50% data	✓
Memory card slot	MMC-Card with max. 1 GB
Hardware configuration	
Racks, max.	1
Modules per rack, max.	8
Number of integrated DP master	0
Number of DP master via CP	4
Operable function modules	8
Operable communication modules PtP	8
Operable communication modules LAN	8
Status information, alarms, diagnostics	
Status display	yes
Interrupts	yes
Process alarm	yes
Diagnostic interrupt	yes
Diagnostic functions	no

Technical data

Order no.	312-5BE13
Diagnostics information read-out	possible
Supply voltage display	green LED
Group error display	red SF LED
Channel error display	red LED per group
Isolation	
Between channels	✓
Between channels of groups to	16
Between channels and backplane bus	✓
Between channels and power supply	-
Max. potential difference between circuits	DC 75 V/ AC 50 V
Max. potential difference between inputs (U _{cm})	-
Max. potential difference between Mana and Mintern (U _{iso})	-
Max. potential difference between inputs and Mana (U _{cm})	-
Max. potential difference between inputs and Mintern (U _{iso})	-
Max. potential difference between Mintern and outputs	-
Insulation tested with	DC 500 V
Command processing times	
Bit instructions, min.	0.02 µs
Word instruction, min.	0.02 µs
Double integer arithmetic, min.	0.02 µs
Floating-point arithmetic, min.	0.12 µs
Timers/Counters and their retentive characteristics	
Number of S7 counters	512
Number of S7 times	512
Data range and retentive characteristic	
Number of flags	8192 Byte
Number of data blocks	4095
Max. data blocks size	64 KB
Max. local data size per execution level	510 Byte
Blocks	
Number of OBs	15
Number of FBs	2048
Number of FCs	2048

Order no.	312-5BE13
Maximum nesting depth per priority class	8
Maximum nesting depth additional within an error OB	4
Time	
Real-time clock buffered	✓
Clock buffered period (min.)	6 w
Accuracy (max. deviation per day)	10 s
Number of operating hours counter	8
Clock synchronization	✓
Synchronization via MPI	Master/Slave
Synchronization via Ethernet (NTP)	no
Address areas (I/O)	
Input I/O address area	1024 Byte
Output I/O address area	1024 Byte
Input process image maximal	128 Byte
Output process image maximal	128 Byte
Digital inputs	272
Digital outputs	264
Digital inputs central	272
Digital outputs central	264
Integrated digital inputs	16
Integrated digital outputs	8
Analog inputs	64
Analog outputs	64
Analog inputs, central	64
Analog outputs, central	64
Integrated analog inputs	0
Integrated analog outputs	0
Communication functions	
PG/OP channel	✓
Global data communication	✓
Number of GD circuits, max.	4
Size of GD packets, max.	22 Byte
S7 basic communication	✓
S7 basic communication, user data per job	76 Byte
S7 communication	✓

Technical data

Order no.	312-5BE13
S7 communication as server	✓
S7 communication as client	-
S7 communication, user data per job	160 Byte
Number of connections, max.	32
PWM data	
PWM channels	2
PWM time basis	0.1ms/1ms
Period length	4...65535 / 1...65535 * timebase
Minimum pulse width	0...0.5*period
Type of output	Highside with 1.1kOhm pulldown
Functionality Sub-D interfaces	
Type	X2
Type of interface	RS485
Connector	Sub-D, 9-pin, female
Electrically isolated	-
MPI	✓
MP ² I (MPI/RS232)	-
DP master	-
DP slave	-
Point-to-point interface	-
Type	X3
Type of interface	RS485
Connector	Sub-D, 9-pin, female
Electrically isolated	✓
MPI	-
MP ² I (MPI/RS232)	-
DP master	-
DP slave	-
Point-to-point interface	✓
Functionality MPI	
Number of connections, max.	32
PG/OP channel	✓
Routing	-
Global data communication	✓
S7 basic communication	✓

Order no.	312-5BE13
S7 communication	✓
S7 communication as server	✓
S7 communication as client	-
Transmission speed, min.	19.2 kbit/s
Transmission speed, max.	187.5 kbit/s
Functionality PROFIBUS master	
PG/OP channel	-
Routing	-
S7 basic communication	-
S7 communication	-
S7 communication as server	-
S7 communication as client	-
Activation/deactivation of DP slaves	-
Direct data exchange (slave-to-slave communication)	-
DPV1	-
Transmission speed, min.	-
Transmission speed, max.	-
Number of DP slaves, max.	-
Address range inputs, max.	-
Address range outputs, max.	-
User data inputs per slave, max.	-
User data outputs per slave, max.	-
Functionality PROFIBUS slave	
PG/OP channel	-
Routing	-
S7 communication	-
S7 communication as server	-
S7 communication as client	-
Direct data exchange (slave-to-slave communication)	-
DPV1	-
Transmission speed, min.	-
Transmission speed, max.	-
Automatic detection of transmission speed	-
Transfer memory inputs, max.	-
Transfer memory outputs, max.	-

Technical data

Order no.	312-5BE13
Address areas, max.	-
User data per address area, max.	-
Point-to-point communication	
PtP communication	✓
Interface isolated	✓
RS232 interface	-
RS422 interface	-
RS485 interface	✓
Connector	Sub-D, 9-pin, female
Transmission speed, min.	150 bit/s
Transmission speed, max.	115.5 kbit/s
Cable length, max.	500 m
Point-to-point protocol	
ASCII protocol	✓
STX/ETX protocol	✓
3964(R) protocol	✓
RK512 protocol	-
USS master protocol	✓
Modbus master protocol	✓
Modbus slave protocol	-
Special protocols	-
Functionality RJ45 interfaces	
Type	X5
Type of interface	Ethernet 10/100 MBit
Connector	RJ45
Electrically isolated	✓
PG/OP channel	✓
Number of connections, max.	4
Productive connections	-
Housing	
Material	PPE
Mounting	Rail System 300
Mechanical data	
Dimensions (WxHxD)	80 mm x 125 mm x 120 mm
Weight	410 g
Environmental conditions	

Order no.	312-5BE13
Operating temperature	0 °C to 60 °C
Storage temperature	-25 °C to 70 °C
Certifications	
UL certification	yes

5 Deployment CPU 312-5BE13

5.1 Assembly



Information about assembly and cabling: ↗ Chapter 3 'Assembly and installation guidelines' on page 16

5.2 Start-up behavior

Turn on power supply

After the power supply has been switched on, the CPU changes to the operating mode the operating mode lever shows.

Default boot procedure, as delivered

When the CPU is delivered it has been reset. After a STOP→RUN transition the CPU switches to RUN without program.

Boot procedure with valid configuration in the CPU

The CPU switches to RUN with the program stored in the battery buffered RAM.

Boot procedure with empty battery

- The accumulator/battery is automatically loaded via the integrated power supply and guarantees a buffer for max. 30 days. If this time is exceeded, the battery may be totally discharged. This means that the battery buffered RAM is deleted.
- In this state, the CPU executes an overall reset. If a MMC is plugged, program code and data blocks are transferred from the MMC into the work memory of the CPU. If no MMC is plugged, the CPU transfers permanent stored "protected" blocks into the work memory if available.
- Depending on the position of the operating mode switch, the CPU switches to RUN, if OB81 exists, res. remains in STOP. This event is stored in the diagnostic buffer as: "Start overall reset automatically (unbuffered PowerON)".



CAUTION!

After a power reset and with an empty battery the CPU starts with a BAT error and executes an overall reset. The BAT error can be deleted again, if once during power cycle the time between switching on and off the power supply is at least 30sec. and the battery is fully loaded. Otherwise with a short power cycle the BAT error still exists and an overall reset is executed.

5.3 Addressing

5.3.1 Overview

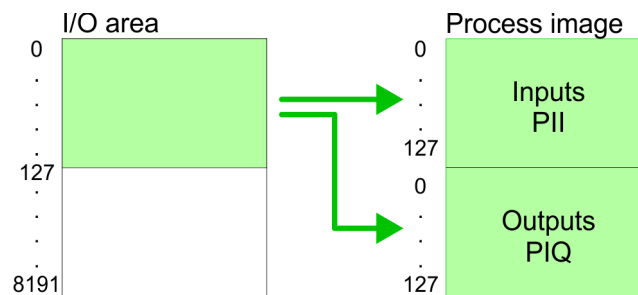
To provide specific addressing of the installed peripheral modules, certain addresses must be allocated in the CPU. At the start-up of the CPU, this assigns automatically peripheral addresses for digital in-/output modules starting with 0 and ascending depending on the slot location. If no hardware project engineering is available, the CPU stores at the addressing analog modules to even addresses starting with 256.

5.3.2 Addressing Backplane bus I/O devices

The CPU 312-5BE13 provides an I/O area (address 0 ... 8191) and a process image of the In- and Outputs (each address 0 ... 127). The process image stores the signal states of the lower address (0 ... 127) additionally in a separate memory area.

The process image this divided into two parts:

- process image to the inputs (PII)
- process image to the outputs (PIQ)



The process image is updated automatically when a cycle has been completed.

Max. number of plug-gable modules

Maximally 8 modules may be addressed by the CPU 312-5BE13. The extension by means of extension racks is not possible.

Define addresses by hardware configuration

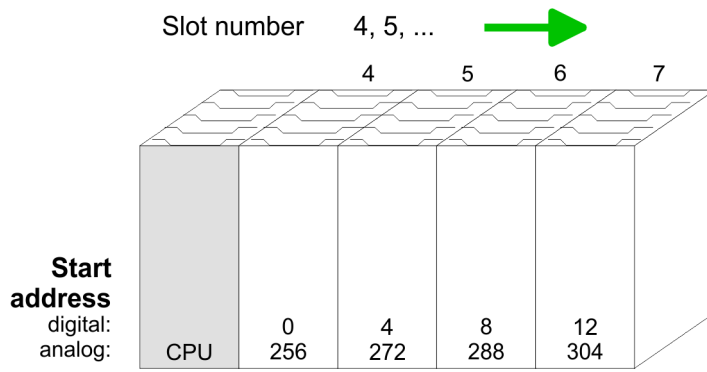
You may access the modules with read res. write accesses to the peripheral bytes or the process image. To define addresses a hardware configuration may be used. For this, click on the properties of the according module and set the wanted address.

Automatic addressing

If you do not like to use a hardware configuration, an automatic addressing comes into force. At the automatic address allocation DIOS occupy depending on the slot location always 4byte and AIOs, FMs, CPs always 16byte at the bus. Depending on the slot location the start address from where on the according module is stored in the address range is calculated with the following formulas:

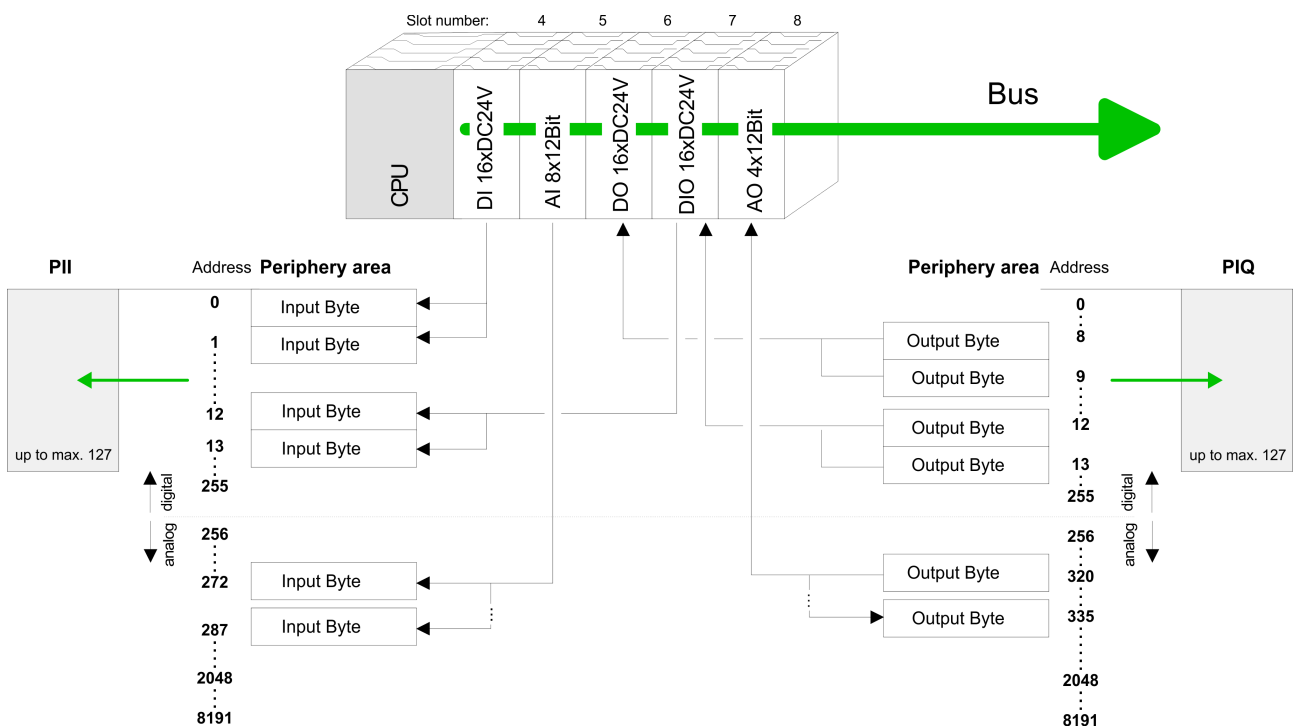
- DIOS: Start address = $4 \times (\text{slot} - 4)$
- AIOs, FMs, CPs: Start address = $16 \times (\text{slot} - 4) + 256$

Address assignment



Example for automatic address allocation

The following sample shows the functionality of the automatic address allocation:



5.4 Address assignment

Input area

Sub module	Default address	Access	Assignment
DI10/DO6	124	Byte	Digital input I+0.0 ... I+0.7
	125	Byte	Digital input I+1.0 ... I+1.7
Counter	768	DInt	Channel 0: Counter value / Frequency value
	772	DInt	Channel 1: Counter value / Frequency value

Sub module	Default address	Access	Assignment
	776	DInt	reserved
	780	DInt	reserved

Output area

Sub module	Default address	Access	Assignment
DI10/DO6	124	Byte	Digital output Q+0.0 ... Q+0.7
Counter	768	DWord	reserved
	772	DWord	reserved
	776	DWord	reserved
	780	DWord	reserved

5.5 Hardware configuration - CPU

Precondition

The configuration of the CPU takes place at the Siemens 'hardware configurator'. The hardware configurator is part of the Siemens SIMATIC Manager. It serves for project engineering. The modules, which may be configured here are listed in the hardware catalog. If necessary you have to update the hardware catalog with 'Options → Update Catalog'.

For project engineering a thorough knowledge of the Siemens SIMATIC Manager and the Siemens hardware configurator is required.



Please consider that this SPEED7-CPU has 4 ACCUs. After an arithmetic operation (+I, -I, *I, /I, +D, -D, *D, /D, MOD, +R, -R, *R, /R) the content of ACCU 3 and ACCU 4 is loaded into ACCU 3 and 2. This may cause conflicts in applications that presume an unmodified ACCU 2.

For more information may be found in the manual "VIPA Operation list SPEED7" at "Differences between SPEED7 and 300V programming".

Proceeding

Slot	Module
1	
2	CPU 312C
2.2	DI10/DO6
2.4	Count
3	

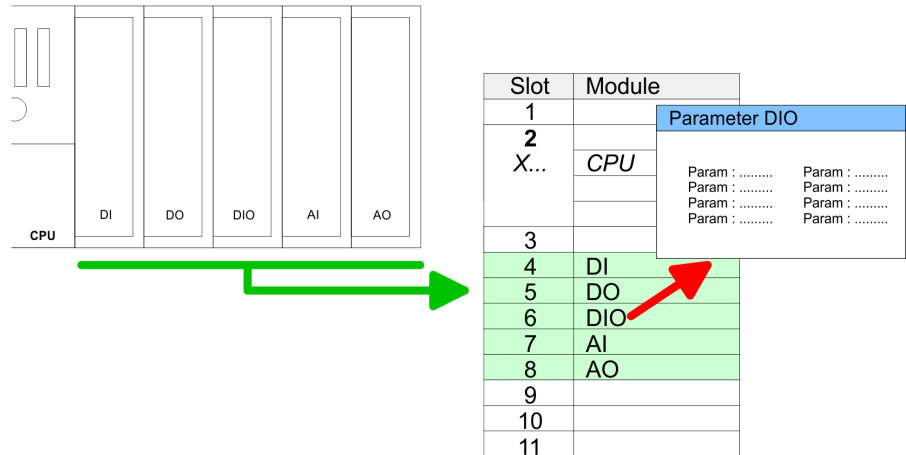
To be compatible with the Siemens SIMATIC Manager the following steps should be executed:

1. ▶ Start the Siemens hardware configurator with a new project.
2. ▶ Insert a profile rail from the hardware catalog.
3. ▶ Place at 'Slot' number 2 the CPU 312C (6ES7 312-5BE03-0AB0 V2.6).

5.6 Hardware configuration - I/O modules

Hardware configuration of the modules

After the hardware configuration place the System 300 modules in the plugged sequence starting with slot 4.



Parametrization

For parametrization double-click during the project engineering at the slot overview on the module you want to parameterize. In the appearing dialog window you may set the wanted parameters. By using the SFCs 55, 56 and 57 you may alter and transfer parameters for wanted modules during runtime. For this you have to store the module specific parameters in so called "record sets". More detailed information about the structure of the record sets is to find in the according module description.

Maximum 8 modules addressable

Maximally 8 modules may be addressed by the CPU 312-5BE13. The extension by means of extension racks is not possible!

5.7 Hardware configuration - Ethernet PG/OP channel

Overview

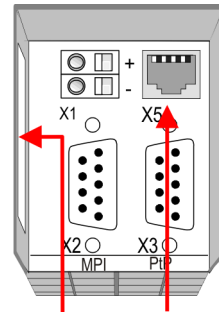
The CPU 312-5BE13 has an integrated Ethernet PG/OP channel. This channel allows you to program and remote control your CPU. The PG/OP channel also gives you access to the internal web page that contains information about firmware version, connected I/O devices, current cycle times etc. With the first start-up respectively after an overall reset the Ethernet PG/OP channel does not have any IP address. For online access to the CPU via Ethernet PG/OP channel valid IP address parameters have to be assigned to this by means of the Siemens SIMATIC Manager. This is called "initialization".

Assembly and commissioning

1. Install your System 300S with your CPU.
2. Wire the system by connecting cables for voltage supply and signals.
3. Connect the Ethernet jack of the Ethernet PG/OP channel to Ethernet
4. Switch on the power supply.
 - ⇒ After a short boot time the CP is ready for communication. He possibly has no IP address data and requires an initialization.

"Initialization" via PLC functions

The initialization via PLC functions takes place with the following proceeding:



**Ethernet PG/OP
address channel**

- ➔ Determine the current Ethernet (MAC) address of your Ethernet PG/OP channel. This always may be found as 1. address under the front flap of the CPU on a sticker on the left side.

Assign IP address parameters

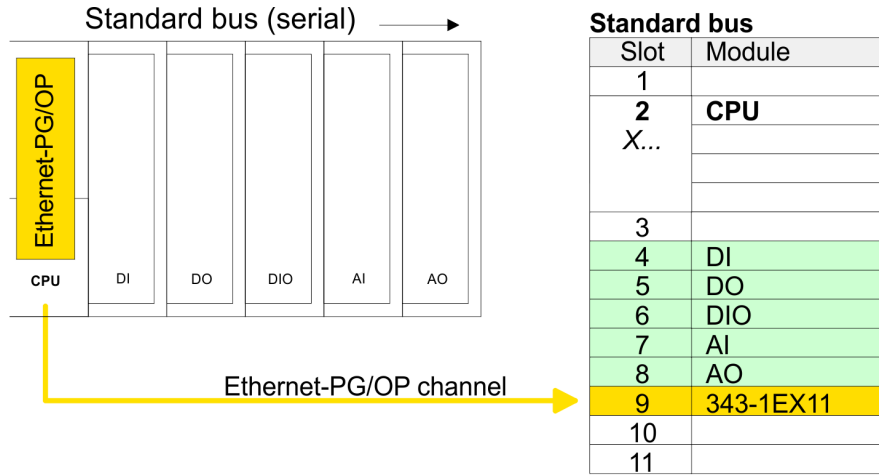
You get valid IP address parameters from your system administrator. The assignment of the IP address data happens online in the Siemens SIMATIC Manager starting with version V 5.3 & SP3 with the following proceeding:

1. ➔ Start the Siemens SIMATIC Manager and set via *'Options → Set PG/PC interface'* the access path to *'TCP/IP -> Network card ...'*.
2. ➔ Open with *'PLC → Edit Ethernet Node n'* the dialog window with the same name.
3. ➔ To get the stations and their MAC address, use the [Browse] button or type in the MAC Address. The Mac address may be found at the 1. label beneath the front flap of the CPU.
4. ➔ Choose if necessary the known MAC address of the list of found stations.
5. ➔ Either type in the IP configuration like IP address, subnet mask and gateway.
6. ➔ Confirm with [Assign IP configuration].
 - ⇒ Direct after the assignment the Ethernet PG/OP channel may be reached online by these address data. The value remains as long as it is reassigned, it is overwritten by a hardware configuration or an factory reset is executed.

Take IP address parameters in project

1. ➔ Open the Siemens hardware configurator und configure the Siemens CPU 312C (6ES7 312-5BE03-0AB0 V2.6).
2. ➔ Configure the modules at the standard bus.
3. ➔ For the Ethernet PG/OP channel you have to configure a Siemens CP 343-1 (SIMATIC 300 \ CP 300 \ Industrial Ethernet \ CP 343-1 \ 6GK7 343-1EX11 0XE0) always below the really plugged modules.
4. ➔ Open the property window via double-click on the CP 343-1EX11 and enter for the CP at *'Properties'* the IP address data, which you have assigned before.
5. ➔ Assign the CP to a *'Subnet'*. Without assignment the IP address data are not used!

6. Transfer your project.



5.8 CPU parametrization

5.8.1 Parametrization via Siemens CPU

Parameterization via Siemens CPU 312C

Since the CPU is to be configured as Siemens CPU 312C (CPU 312-5BE03-0AB0 V2.6) in the Siemens hardware configurator, the parameters of the CPU 312-5BE13 may be set with "Object properties" of the CPU 312C during hardware configuration. Via a double-click on the CPU 312C the parameter window of the CPU may be accessed. Using the registers you get access to every parameter of the CPU.

Slot	Module
1	
2	CPU ...
...	...
3	
4	

Parameter CPU	
Param :	Param :
Param :	Param :
Param :	Param :
Param :	Param :



Description of the parameters of the sub modules 'DI10/DO6' and 'Counter' & Chapter 6 'Deployment I/O periphery' on page 90

5.8.2 CPU parameters

Supported parameters

The CPU does not evaluate each parameter, which may be set at the hardware configuration. The following parameters are supported by the CPU at this time:

General

- Short description: The short description of the Siemens CPU 312-5BE03 is CPU 312C.
- Order No. / Firmware: Order number and firmware are identical to the details in the "hardware catalog" window.

- Name: The Name field provides the short description of the CPU. If you change the name the new name appears in the Siemens SIMATIC Manager.
- Interface: Here is the address of the MPI interface.
- Properties: By means of this button you can change the properties of the MPI interface.
- Comment: In this field information about the module may be entered.

Startup

- Startup when expected/actual configuration differs: If the checkbox for *'Startup when expected/actual configuration differ'* is deselected and at least one module is not located at its configured slot or if another type of module is inserted there instead, then the CPU does not switch to RUN mode and remains in STOP mode. If the checkbox for *'Startup when expected/actual configuration differ'* is selected, then the CPU starts even if there are modules not located in their configured slots or if another type of module is inserted there instead, such as during an initial system start-up.
- Monitoring time for ready message by modules [100ms]: This operation specifies the maximum time for the ready message of every configured module after PowerON. If the modules do not send a ready message to the CPU by the time the monitoring time has expired, the actual configuration becomes unequal to the preset configuration.
- Transfer of parameters to modules [100ms]: The maximum time for the transfer of parameters to parameterizable modules. If not every module has been assigned parameters by the time this monitoring time has expired; the actual configuration becomes unequal to the preset configuration.

Cycle/Clock memory

- Update OB1 process image cyclically: This parameter is not relevant.
- Scan cycle monitoring time: Here the scan cycle monitoring time in milliseconds may be set. If the scan cycle time exceeds the scan cycle monitoring time, the CPU enters the STOP mode. Possible reasons for exceeding the time are:
 - Communication processes
 - a series of interrupt events
 - an error in the CPU program
- Minimum scan cycle time: This parameter is not relevant.
- Scan cycle load from Communication: This parameter is not relevant.
- Size of the process image input/output area: Here the size of the process image max. 2048 for the input/output periphery may be fixed.
- OB85 call up at I/O access error: The preset reaction of the CPU may be changed to an I/O access error that occurs during the update of the process image by the system. The VIPA CPU is preset such that OB 85 is not called if an I/O access error occurs and no entry is made in the diagnostic buffer either.
- Clock memory: Activate the check box if you want to use clock memory and enter the number of the memory byte.



The selected memory byte cannot be used for temporary data storage.

- Retentive Memory**
- Number of Memory bytes from MB0: Enter the number of retentive memory bytes from memory byte 0 onwards.
 - Number of S7 Timers from T0: Enter the number of retentive S7 timers from T0 onwards. Each S7 timer occupies 2bytes.
 - Number of S7 Counters from C0: Enter the number of retentive S7 counter from C0 onwards.
 - Areas: This parameter is not supported.
- Interrupts**
- Priority: Here the priorities are displayed, according to which the hardware interrupt OBs are processed (hardware interrupt, time-delay interrupt, async. error interrupts).
- Time-of-day interrupts**
- Priority: The priority may not be modified.
 - Active: Activate the check box of the time-of-day interrupt OBs if these are to be automatically started on complete restart.
 - Execution: Select how often the interrupts are to be triggered. Intervals ranging from every minute to yearly are available. The intervals apply to the settings made for *start date* and *time*.
 - Start date/time: Enter date and time of the first execution of the time-of-day interrupt.
 - Process image partition: This parameter is not supported.
- Cyclic interrupts**
- Priority: Here the priorities may be specified according to which the corresponding cyclic interrupt is processed. With priority "0" the corresponding interrupt is deactivated.
 - Execution: Enter the time intervals in ms, in which the watchdog interrupt OBs should be processed. The start time for the clock is when the operating mode switch is moved from STOP to RUN.
 - Phase offset: Enter the delay time in ms for current execution for the watch dog interrupt. This should be performed if several watchdog interrupts are enabled. Phase offset allows to distribute processing time for watchdog interrupts across the cycle.
 - Process image partition: This parameter is not supported.
- Protection**
- Level of protection: Here 1 of 3 protection levels may be set to protect the CPU from unauthorized access.
 - *Protection level 1 (default setting):*
No password adjustable, no restrictions
 - *Protection level 2 with password:*
Authorized users: read and write access
Unauthorized user: read access only
 - *Protection level 3:*
Authorized users: read and write access
Unauthorized user: no read and write access

5.9 Project transfer

Overview

There are the following possibilities for project transfer into the CPU:

- Transfer via MPI
- Transfer via Ethernet
- Transfer via MMC

5.9.1 Transfer via MPI

General

For transfer via MPI/PROFIBUS there is the following interface:

- X2: MPI interface

Net structure

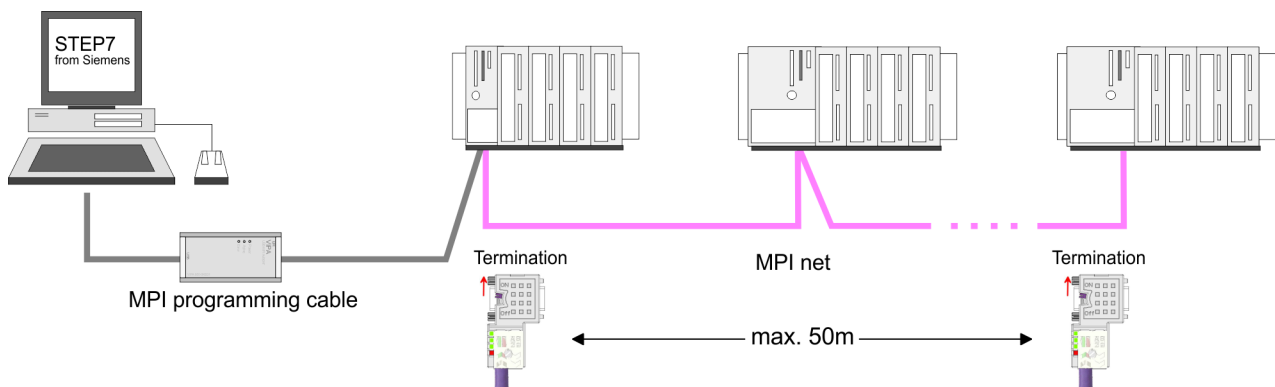
The structure of a MPI net is electrically identical with the structure of a PROFIBUS net. This means the same rules are valid and you use the same components for the build-up. The single participants are connected with each other via bus interface plugs and PROFIBUS cables. Please consider with the CPU 312-5BE13 that the total extension of the MPI net does not exceed 50m. Per default the MPI net runs with 187.5kbaud. VIPA CPUs are delivered with MPI address 2.

MPI programming cable

The MPI programming cables are available at VIPA in different variants. The cables provide a RS232 res. USB plug for the PC and a bus enabled RS485 plug for the CPU. Due to the RS485 connection you may plug the MPI programming cables directly to an already plugged plug on the RS485 jack. Every bus participant identifies itself at the bus with a unique address, in the course of the address 0 is reserved for programming devices.

Terminating resistor

A cable has to be terminated with its surge impedance. For this you switch on the terminating resistor at the first and the last participant of a network or a segment. Please make sure that the participants with the activated terminating resistors are always power supplied. Otherwise it may cause interferences on the bus.



Approach transfer via MPI interface

1. ▶ Connect your PC to the MPI jack of your CPU via a MPI programming cable.
2. ▶ Load your project in the SIMATIC Manager from Siemens.
3. ▶ Choose in the menu '*Options → Set PG/PC interface*'.
4. ▶ Select in the according list the "PC Adapter (MPI)"; if appropriate you have to add it first, then click on [Properties].
5. ▶ Set in the register MPI the transfer parameters of your MPI net and type a valid *address*.
6. ▶ Switch to the register *Local connection*.
7. ▶ Set the COM port of the PCs and the transfer rate 38400Baud for the MPI programming cable from VIPA.
8. ▶ Via '*PLC → Load to module*' via MPI to the CPU and save it on a MMC via '*PLC → Copy RAM to ROM*' if one is plugged.

5.9.2 Transfer via Ethernet

For transfer via Ethernet the CPU has the following interface:

- X5: Ethernet PG/OP channel

Initialization

So that you may access the Ethernet PG/OP channel you have to assign IP address parameters by means of the "initialization".
 ↪ *'Hardware configuration - Ethernet PG/OP channel' on page 50*

Transfer

1. ➤ For the transfer, connect, if not already done, the appropriate Ethernet port to your Ethernet.
2. ➤ Open your project with the Siemens SIMATIC Manager.
3. ➤ Set via *'Options → Set PG/PC Interface'* the access path to "TCP/IP → Network card".
4. ➤ Click to *'PLC → Download'* Download → the dialog "Select target module" is opened. Select your target module and enter the IP address parameters of the Ethernet PG/OP channel for connection. Provided that no new hardware configuration is transferred to the CPU, the entered Ethernet connection is permanently stored in the project as transfer channel.
5. ➤ With [OK] the transfer is started.



System dependent you get a message that the projected system differs from target system. This message may be accepted by [OK].

→ Your project is transferred and may be executed in the CPU after transfer.

5.9.3 Transfer via MMC

The MMC (**M**emory **C**ard) serves as external transfer and storage medium. There may be stored several projects and sub-directories on a MMC storage module. Please regard that your current project is stored in the root directory and has one of the following file names:

- S7PROG.WLD
- AUTOLOAD.WLD

With *'File → Memory Card File → New'* in the Siemens SIMATIC Manager a new wld file may be created. After the creation copy the blocks from the project blocks folder and the System data into the wld file.

Transfer MMC → CPU

The transfer of the application program from the MMC into the CPU takes place depending on the file name after an overall reset or PowerON.

- *S7PROG.WLD* is read from the MMC after overall reset.
- *AUTOLOAD.WLD* is read after PowerON from the MMC.

The blinking of the MC LED of the CPU marks the active transfer. Please regard that your user memory serves for enough space, otherwise your user program is not completely loaded and the SF LED gets on.

Transfer CPU → MMC

When the MMC has been installed, the write command stores the content of the battery buffered RAM as S7PROG.WLD on the MMC.

The write command is controlled by means of the block area of the Siemens SIMATIC Manager 'PLC → Copy RAM to ROM'. During the write process the MC LED of the CPU is blinking. When the LED expires the write process is finished.

If this project is to be loaded automatically from the MMC with PowerON, you have to rename this on the MMC to *AUTOLOAD.WLD*.

Transfer control

After a MMC access, an ID is written into the diagnostic buffer of the CPU. To monitor the diagnosis entries, you select 'PLC → Module Information' in the Siemens SIMATIC Manager. Via the register "Diagnostic Buffer" you reach the diagnosis window.

Information about the event IDs ↪ *Chapter 5.19 'VIPA specific diagnostic entries' on page 72.*

5.10 Access to the internal Web page

Access to the web page

The Ethernet PG/OP channel provides a web page that you may access via an Internet browser by its IP address. The web page contains information about firmware versions, current cycle times etc. The current content of the web page is stored on MMC by means of the MMC-Cmd *WEBPAGE*. ↪ *Chapter 5.18 'MMC-Cmd - Auto commands' on page 70*

Requirements

A PG/OP channel connection should be established between PC with Internet browser and CPU 312-5BE13. This may be tested by Ping to the IP address of the PG/OP channel.

Access to the internal Web page

Web page

The access takes place via the IP address of the Ethernet PG/OP channel. The web page only serves for information output. The monitored values are not alterable.



CPU with Ethernet-PG/OP

Slot 100	
VIPA 312-5BE13 V.... Px000135.pkg, SERIALNUMBER 05439	Order no., firmware vers., package, serial no.
SUPPORTDATA : PRODUCT V3420, HARDWARE V0114, 5679H-V20 , HX000027.110 , Bx000227 V6420, Ax000086 V1200, Ax000056 V0220, fx000007.wld V1140, FlashFileSystem : V102	Information for support
Memorysize(Bytes):LoadMem:LoadMem : 524288, Work- MemCode : 26214, WorkMemData : 26214	Information about memory con- figuration, load memory, work memory (code/data)
OnBoardEthernet : MacAddress : 0020D50144C1, IP- Address : 172.20.120.62, SubnetMask : 255.255.255.0, Gateway : 172.20.120.62	Ethernet PG/OP: Addresses
Cpu state : Run	CPU state
FunctionRS485 X2/COM1: MPI FunctionRS485 X3/COM2: PtP	Operating mode RS485 PtP: point to point operation
Cycletime [microseconds] : min=0 cur=770 ave=750 max=878	CPU cycle time: min= minimal cur= current max= maximal
MCC-Trial-Time: 70:23	Remaining time in hh:mm for deactivation of the expansion memory if MCC is removed.
ArmLoad [percent] : cur=67, max=70	Information for support
PowerCycleHxRetries : 29, 0, 0, 0, 0	
AutoCompress activated	

Slot 202	
VIPA DI16/DO8 V3.2.9,SUPPORTDATA:PRODUCT...	CPU component: Digital I/Os Name, firmware version, module type
SUPPORTDATA: PRODUCT V3290, Module Type ...	Information for support

Slot 202		CPU component: Digital I/Os
Address Input 124...125		Configured input base addresses
Address Output 124...124		Configured output base addresses

Slot 204		CPU component: Counter
VIPA 2 COUNTERS V3.2.9,		Name, firmware version, module type
SUPPORTDATA: PRODUCT V3290, Module Type ...		Information for support
Address Input 768...783		Configured input base addresses
Address Output 768...783		Configured output base addresses

Standard Bus

Standard Bus		Modules at the standard bus
8 Bit Mode		Information for support

5.11 Operating modes

5.11.1 Overview

The CPU can be in one of 4 operating modes:

- Operating mode STOP
- Operating mode START-UP
- Operating mode RUN
- Operating mode HALT

Certain conditions in the operating modes START-UP and RUN require a specific reaction from the system program. In this case the application interface is often provided by a call to an organization block that was included specifically for this event.

Operating mode STOP

- The application program is not processed.
- If there has been a processing before, the values of counters, timers, flags and the process image are retained during the transition to the STOP mode.
- Outputs are inhibited, i.e. all digital outputs are disabled.
- RUN-LED off
- STOP-LED on

Operating mode START-UP

- During the transition from STOP to RUN a call is issued to the start-up organization block OB 100. The processing time for this OB is not monitored. The START-UP OB may issue calls to other blocks.
- All digital outputs are disabled during the START-UP, i.e. outputs are inhibited.

- RUN-LED
blinks as soon as the OB 100 is operated and for at least 3s, even if the start-up time is shorter or the CPU gets to STOP due to an error. This indicates the start-up.
- STOP-LED off

When the CPU has completed the START-UP OB, it assumes the operating mode RUN.

Operating mode RUN

- The application program in OB 1 is processed in a cycle. Under the control of alarms other program sections can be included in the cycle.
- All timers and counters being started by the program are active and the process image is updated with every cycle.
- The BASP-signal (outputs inhibited) is deactivated, i.e. all digital outputs are enabled.
- RUN-LED on
- STOP-LED off

Operating mode HOLD

The CPU offers up to 3 breakpoints to be defined for program diagnosis. Setting and deletion of breakpoints happens in your programming environment. As soon as a breakpoint is reached, you may process your program step by step.

Precondition

For the usage of breakpoints, the following preconditions have to be fulfilled:

- Testing in single step mode is possible with STL. If necessary switch the view via 'View → STL' to STL.
- The block must be opened online and must not be protected.

Approach for working with breakpoints

1. ▶ Activate 'View → Breakpoint Bar'.
2. ▶ Set the cursor to the command line where you want to insert a breakpoint.
3. ▶ Set the breakpoint with 'Debug → Set Breakpoint'.
⇒ The according command line is marked with a circle.
4. ▶ To activate the breakpoint click on 'Debug → Breakpoints Active'.
⇒ The circle is changed to a filled circle.
5. ▶ Bring your CPU into RUN.
⇒ When the program reaches the breakpoint, your CPU switches to the state HOLD, the breakpoint is marked with an arrow and the register contents are monitored.
6. ▶ Now you may execute the program code step by step via 'Debug → Execute Next Statement' or run the program until the next breakpoint via 'Debug → Resume'.
7. ▶ Delete (all) breakpoints with the option 'Debug → Delete All Breakpoints'.

Behavior in operating state HOLD

- The RUN-LED blinks and the STOP-LED is on.
- The execution of the code is stopped. No level is further executed.

- All times are frozen.
- The real-time clock runs is just running.
- The outputs were disabled (BASP is activated).
- Configured CP connections remain exist.



The usage of breakpoints is always possible. Switching to the operating mode test operation is not necessary.
With more than 2 breakpoints, a single step execution is not possible.

5.11.2 Function security

The CPUs include security mechanisms like a Watchdog (100ms) and a parametrizable cycle time surveillance (parametrizable min. 1ms) that stop res. execute a RESET at the CPU in case of an error and set it into a defined STOP state. The VIPA CPUs are developed function secure and have the following system properties:

Event	concerns	Effect
RUN → STOP	general	BASP (Befehls-Ausgabe-Sperre, i.e. command output lock) is set.
	central digital outputs	The outputs are disabled.
	central analog outputs	The outputs are disabled. <ul style="list-style-type: none"> ■ Voltage outputs issue 0V ■ Current outputs 0...20mA issue 0mA ■ Current outputs 4...20mA issue 4mA If configured also substitute values may be issued.
	decentral outputs	Same behavior as the central digital/analog outputs.
	decentral inputs	The inputs are cyclically be read by the decentralized station and the recent values are put at disposal.
STOP → RUN res. PowerON	general	First the PII is deleted, then OB 100 is called. After the execution of the OB, the BASP is reset and the cycle starts with: Delete PIO → Read PII → OB 1.
	decentral inputs	The inputs are once be read by the decentralized station and the recent values are put at disposal.
RUN	general	The program execution happens cyclically and can therefore be foreseen: Read PII → OB 1 → Write PIO.

PII: Process image inputs, PIO: Process image outputs

5.12 Overall reset

Overview

During the overall reset the entire user memory is erased. Data located in the memory card is not affected. You have 2 options to initiate an overall reset:

- initiate the overall reset by means of the operating mode switch
- initiate the overall reset by means of the Siemens SIMATIC Manager



You should always issue an overall reset to your CPU before loading an application program into your CPU to ensure that all blocks have been cleared from the CPU.

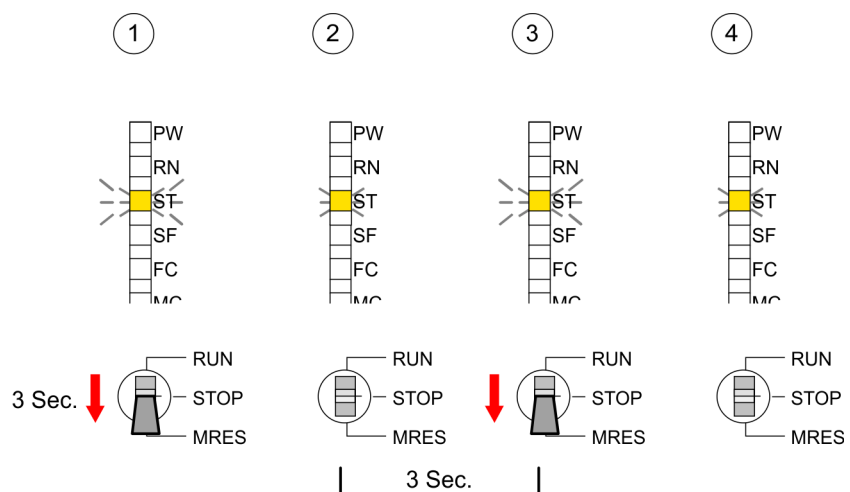
Overall reset by means of the operating mode switch

Precondition

- ➔ The operating mode of the CPU is to be switched to STOP. For this switch the operating mode switch of the CPU to "STOP".
 - ⇒ The STOP-LED is on.

Overall reset

- 1.** ➔ Switch the operating mode switch to MRES position for about 3 seconds.
 - ⇒ The STOP-LED changes from blinking to permanently on.
- 2.** ➔ Place the operating mode switch in the position STOP and switch it to MRES and quickly back to STOP within a period of less than 3 seconds.
 - ⇒ The STOP-LED blinks (overall reset procedure).
- 3.** ➔ The overall reset has been completed when the STOP-LED is on permanently.
 - ⇒ The STOP-LED is on. The following figure illustrates the above procedure:



Overall reset by means of the Siemens SIMATIC Manager

- Precondition The operating mode of the CPU is to be switched to STOP. You may place the CPU in STOP by the menu command 'PLC → Operating mode'.
- Overall reset: You may request the overall reset by means of the menu command 'PLC → Clean/Reset'. In the dialog window you may place your CPU in STOP state and start the overall reset if this has not been done as yet. The STOP-LED blinks during the overall reset procedure. When the STOP-LED is on permanently the overall reset procedure has been completed.

Automatic reload

If there is a project S7PROG.WLD on the MMC, the CPU attempts to reload this project from MMC.

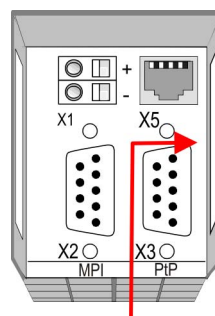
→ The MC LED is on. When the reload has been completed the LED expires. The operating mode of the CPU will be STOP respectively RUN, depending on the position of the operating mode switch.

Reset to factory setting

The *Reset to factory setting* deletes completely the internal RAM of the CPU and resets this to delivery state. Please regard that the MPI address is also set back to default 2! ↪ *Chapter 5.14 'Reset to factory setting' on page 66*

5.13 Firmware update**Overview**

- There is the opportunity to execute a firmware update for the CPU and its components via MMC. For this an accordingly prepared MMC must be in the CPU during the startup.
- So a firmware files can be recognized and assigned with startup, a pkg file name is reserved for each updateable component an hardware release, which begins with "px" and differs in a number with six digits. The pkg file name of every updateable component may be found at a label right down the front flap of the module.
- After PowerON and CPU STOP the CPU checks if there is a *.pkg file on the MMC. If this firmware version is different to the existing firmware version, this is indicated by blinking of the LEDs and the firmware may be installed by an update request.



Firmware package and version

Latest firmware at www.vipa.com

The latest firmware versions are to be found in the service area at www.vipa.com. For example the following files are necessary for the firmware update of the CPU 312-5BE13 and its components with hardware release 1:

- 312-5BE13, Hardware release 1: Px000135.pkg



CAUTION!

When installing a new firmware you have to be extremely careful. Under certain circumstances you may destroy the CPU, for example if the voltage supply is interrupted during transfer or if the firmware file is defective. In this case, please call the VIPA-Hotline!

Please regard that the version of the update firmware has to be different from the existing firmware otherwise no update is executed.

Display the Firmware version of the SPEED7 system via Web Site

The CPU has an integrated website that monitors information about firmware version of the SPEED7 components. The Ethernet PG/OP channel provides the access to this web site. The CPU has an integrated website that monitors information about firmware version of the SPEED7 components. The Ethernet PG/OP channel provides the access to this web site 'PLC → Assign Ethernet Address'. After that you may access the PG/OP channel with a web browser via the IP address of the project engineering.

↳ Chapter 5.10 'Access to the internal Web page' on page 57

Determine CPU firm-ware version with module information

1. ▶ First establish an online connection to the CPU.
2. ▶ To show the module information you have to select 'PLC → Module information' in the Siemens SIMATIC Manager.
3. ▶ Via the register 'General' the window with hardware and firm-ware version may be selected.

⇒ Due to software-technical reasons there is something different of the VIPA CPU 312-5BE13 to the CPU 312C from Siemens:

Description:	CPU 312C	System identification:	SIMATIC 300									
Name:	CPU 312C											
Version:	<table border="1"> <thead> <tr> <th>Order No./Description</th> <th>Component</th> <th>Version</th> </tr> </thead> <tbody> <tr> <td>6ES7 312-5BE03</td> <td>Hardware</td> <td>1</td> </tr> <tr> <td>VIPA 312-5BE13-0100</td> <td>Firmware</td> <td>V3.6.0</td> </tr> </tbody> </table>			Order No./Description	Component	Version	6ES7 312-5BE03	Hardware	1	VIPA 312-5BE13-0100	Firmware	V3.6.0
Order No./Description	Component	Version										
6ES7 312-5BE03	Hardware	1										
VIPA 312-5BE13-0100	Firmware	V3.6.0										

1

2 3

4

- 1 VIPA order no. (VIPA 312-5BE13)
- 2 Hardware release (01)
- 3 Internal hardware version (00)
- 4 Firmware version (V3.6.0)



Every register of the module information dialog is supported by the VIPA CPUs. More about these registers may be found in the online help of the Siemens SIMATIC manager.

Load firmware and transfer it to MMC

- Go to www.vipa.com
- Click on 'Service → Download → Firmware'.
- Navigate via 'System 300S → CPU' to your CPU and download the zip file to your PC.
- Extract the zip file and copy the extracted pkg files to your MMC.



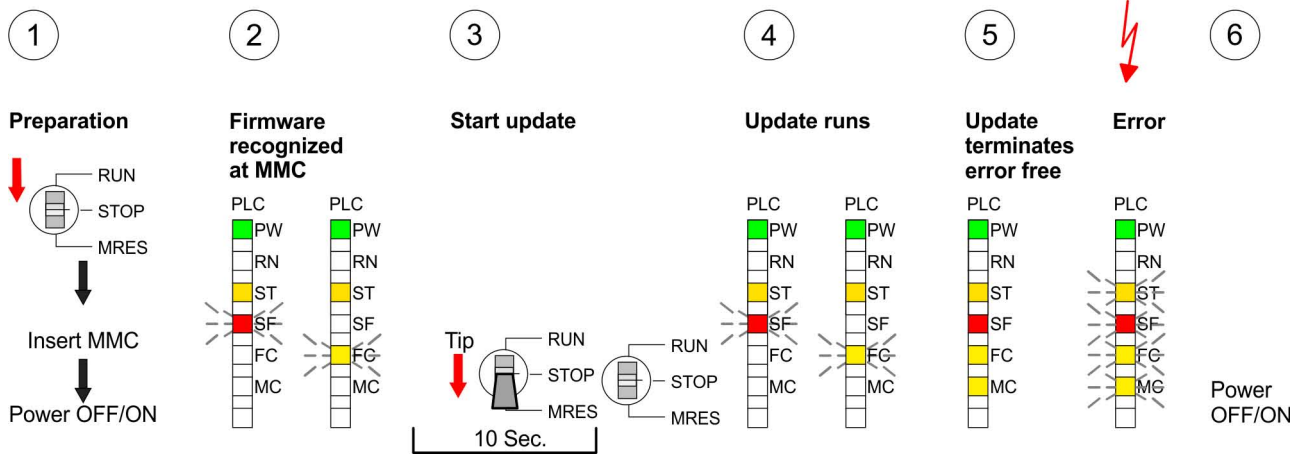
CAUTION!

With a firmware update an overall reset is automatically executed. If your program is only available in the load memory of the CPU it is deleted! Save your program before executing a firmware update! After the firmware update you should execute a "Set back to factory settings". ↪ [Chapter 5.14 'Reset to factory setting' on page 66](#)

Transfer firmware from MMC into CPU

1. ▶ Switch the operating mode switch of your CPU in position STOP. Turn off the voltage supply. Plug the MMC with the firmware files into the CPU. Please take care of the correct plug-in direction of the MMC. Turn on the voltage supply.
2. ▶ After a short boot-up time, the alternate blinking of the LEDs SF and FC shows that at least a more current firmware file was found on the MMC.
3. ▶ You start the transfer of the firmware as soon as you tip the operating mode switch downwards to MRES within 10s.
4. ▶ During the update process, the LEDs SF and FC are alternately blinking and MC LED is on. This may last several minutes.
5. ▶ The update is successful finished when the LEDs PW, ST, SF, FC and MC are on. If they are blinking fast, an error occurred.

6. Turn Power OFF and ON. Now it is checked by the CPU, whether further current firmware versions are available at the MMC. If so, again the LEDs SF and FC flash after a short start-up period. Continue with point 3.
 - ⇒ If the LEDs do not flash, the firmware update is ready. Now a *factory reset* should be executed (see next page). After that the CPU is ready for duty.



5.14 Reset to factory setting

Proceeding

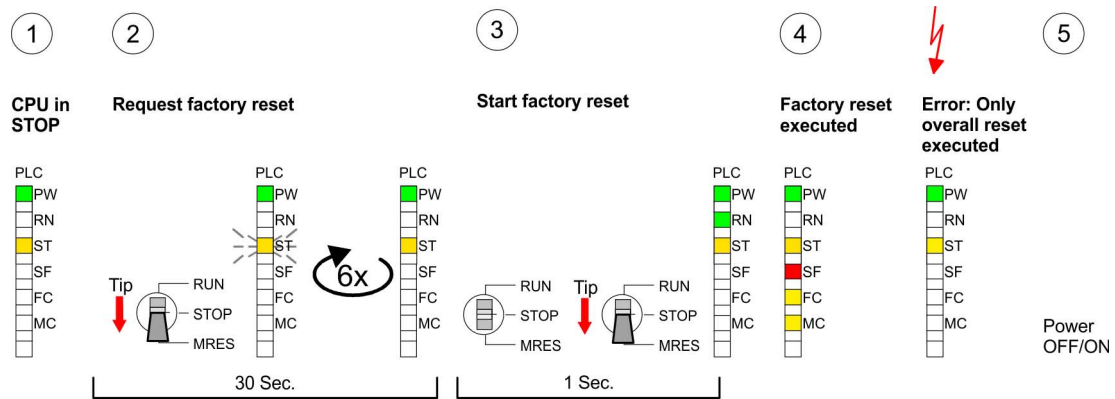
With the following proceeding the internal RAM of the CPU is completely deleted and the CPU is reset to delivery state.

Please note that here also the IP address of the Ethernet PG/OP channel is set to 0.0.0.0 and the MPI address is reset to the address 2!

A reset to factory setting may also be executed by the MMC-*Cmd* `FACTORY_RESET`. ↪ *Chapter 5.18 'MMC-Cmd - Auto commands' on page 70*

1. Switch the CPU to STOP.
2. Push the operating mode switch down to position MRES for 30s. Here the STOP-LED flashes. After a few seconds the stop LED changes to static light. Now the STOP LED changes between static light and flashing. Starting here count the static light states.
3. After the 6. static light release the operating mode switch and tip it downwards to MRES. Now the RUN LED lights up once. This means that the RAM was deleted completely.
4. For the confirmation of the resetting procedure the LEDs PW, ST, SF, FC and MC get ON. If not, the factory reset has failed and only an overall reset was executed. In this case you can repeat the procedure. A factory reset can only be executed if the stop LED has static light for exactly 6 times.
5. The end of factory reset is shown by static light of the LEDs PW, ST, SF, FC and MC. Switch the power supply off and on.

The proceeding is shown in the following Illustration:



After the firmware update you always should execute a Reset to factory setting.

5.15 Slot for storage media

Overview

At the front of the CPU there is a slot for storage media. As external storage medium for applications and firmware you may use a multi-media card (MMC). You can cause the CPU to load a project automatically respectively to execute a command file by means of pre-defined file names.

Accessing the storage medium

To the following times an access takes place on a storage medium:

- After overall reset
 - The CPU checks if there is a project S7PROG.WLD. If exists the project is automatically loaded.
 - The CPU checks if there is a project PROTECT.WLD with protected blocks. If exists the project is automatically loaded. These blocks are stored in the CPU until the CPU is reset to factory setting or an empty PROTECT.WLD is loaded
 - The CPU checks if a MCC memory extension card is put. If exists the memory extension is enabled, otherwise a memory expansion, which was activated before, is de-activated.
- After PowerON
 - The CPU checks if there is a project AUTOLOAD.WLD. If exists an overall reset is established and the project is automatically loaded.
 - The CPU checks if there is a command file with VIPA_CMD.MMC. If exists the command file is loaded and the containing instructions are executed.
 - After PowerON and CPU STOP the CPU checks if there is a *.pkg file (firmware file). If exists this is indicated by blinking of the LEDs and the firmware may be installed by an update request.
- Once in STOP
 - If a storage medium is put, which contains a command file VIPA_CMD.MMC, the command file is loaded and the containing instructions are executed.

5.16 Memory extension with MCC

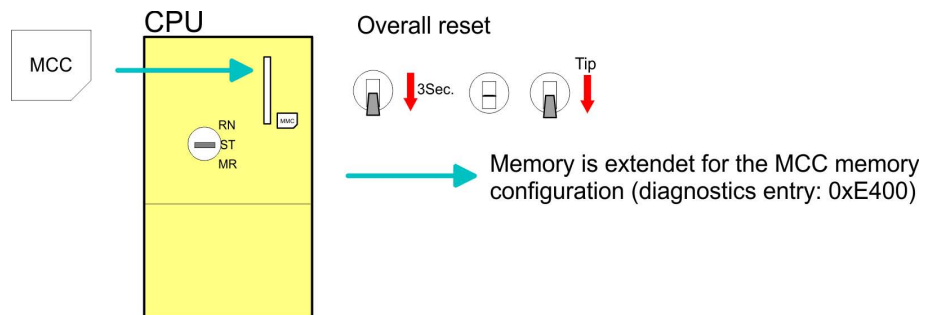
Overview



There is the possibility to extend the work memory of the CPU. For this, a MCC memory extension card is available from VIPA. The MCC is a specially prepared MMC (Multimedia Card). By plugging the MCC into the MCC slot and then an overall reset the according memory expansion is released. There may only one memory expansion be activated at one time. On the MCC there is the file memory.key. This file may not be altered or deleted. You may use the MCC also as "normal" MMC for storing your project.

Proceeding

To extend the memory, plug the MCC into the card slot at the CPU labelled with "MCC" and execute an overall reset.



If the memory expansion on the MCC exceeds the maximum extendible memory range of the CPU, the maximum possible memory of the CPU is automatically used. You may determine the recent memory extension via the integrated web page or with the Siemens SIMATIC Manager at Module Information - "Memory".



CAUTION!

Please regard that the MCC must remain plugged when you've executed the memory expansion at the CPU. Otherwise the CPU switches to STOP after 72 hours. The MCC cannot be exchanged with a MCC of the same memory configuration.

Behavior

When the MCC memory configuration has been taken over you may find the diagnostic entry 0xE400 in the diagnostic buffer of the CPU.

After pulling the MCC the entry 0xE401 appears in the diagnostic buffer, the SF LED is on and after 72 hours the CPU switches to STOP. A reboot is only possible after plugging-in the MCC again or after an overall reset.

The remaining time after pulling the MCC is always been shown with the parameter *MCC-Trial-Time* on the web page.

After re-plugging the MCC, the SF LED extinguishes and 0xE400 is entered into the diagnostic buffer. You may reset the memory configuration of your CPU to the initial status at any time by executing an overall reset without MCC.

5.17 Extended know-how protection

Overview

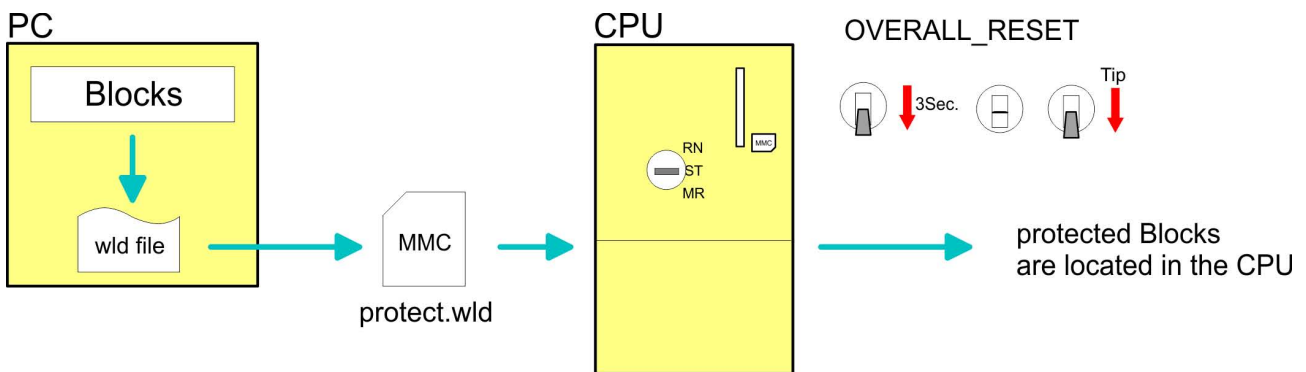
Besides the "standard" Know-how protection the SPEED7-CPU's from VIPA provide an "extended" know-how protection that serves a secure block protection for accesses of 3. persons.

Standard protection

The standard protection from Siemens transfers also protected blocks to the PG but their content is not displayed. But with according manipulation the Know-how protection is not guaranteed.

Extended protection

The "extended" know-how protection developed by VIPA offers the opportunity to store blocks permanently in the CPU. At the "extended" protection you transfer the protected blocks into a WLD-file named protect.wld. By plugging the MMC and following overall reset, the blocks in the protect.wld are permanently stored in the CPU. You may protect OBs, FBs and FCs. When back-reading the protected blocks into the PG, exclusively the block header are loaded. The block code that is to be protected remains in the CPU and cannot be read.

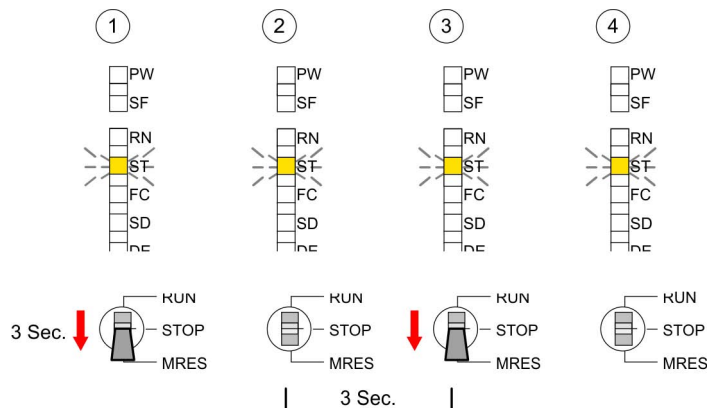


Protect blocks with protect.wld

Create a new wld-file in your project engineering tool with 'File → Memory Card file → New' and rename it to "protect.wld". Transfer the according blocks into the file by dragging them with the mouse from the project to the file window of protect.wld.

Transfer protect.wld to CPU with overall reset

Transfer the file protect.wld to a MMC storage module, plug the MMC into the CPU and execute an overall reset with the following approach:



The overall reset stores the blocks in protect.wld permanently in the CPU protected from accesses of 3. persons.

- Protection behavior** Protected blocks are overwritten by a new protect.wld. Using a PG 3. persons may access protected blocks but only the block header is transferred to the PG. The block code that is to be protected remains in the CPU and cannot be read.
- Change respectively delete protected blocks** Protected blocks in the RAM of the CPU may be substituted at any time by blocks with the same name. This change remains up to next overall reset. Protected blocks may permanently be overwritten only if these are deleted at the protect.wld before. By transferring an empty protect.wld from the MMC you may delete all protected blocks in the CPU.
- Usage of protected blocks** Due to the fact that reading of a "protected" block from the CPU monitors no symbol labels it is convenient to provide the "block covers" for the end user. For this, create a project out of all protected blocks. Delete all networks in the blocks so that these only contain the variable definitions in the according symbolism.

5.18 MMC-Cmd - Auto commands

Overview A *command* file at a MMC is automatically executed under the following conditions:

- CPU is in STOP and MMC is stuck
- After each PowerON

Command file The *command* file is a text file, which consists of a command sequence to be stored as **vipa_cmd.mmc** in the root directory of the MMC. The file has to be started by CMD_START as 1. command, followed by the desired commands (no other text) and must be finished by CMD_END as last command.

Text after the last command CMD_END e.g. comments is permissible, because this is ignored. As soon as the command file is recognized and executed each action is stored at the MMC in the log file logfile.txt. In addition for each executed command a diagnostics entry may be found in the diagnostics buffer.

Commands Please regard the command sequence is to be started with *CMD_START* and ended with *CMD_END*.

Command	Description	Diagnostics entry
CMD_START	In the first line CMD_START is to be located.	0xE801
	There is a diagnostic entry if CMD_START is missing	0xE8FE
WAIT1SECOND	Waits about 1 second.	0xE803
WEBPAGE	The current web page of the CPU is stored at the MMC as" webpage.htm".	0xE804
LOAD_PROJECT	The function "Overall reset and reload from MMC" is executed. The wld file located after the command is loaded else "s7prog.wld" is loaded.	0xE805

Command	Description	Diagnostics entry
SAVE_PROJECT	The recent project (blocks and hardware configuration) is stored as "s7prog.wld" at the MMC. If the file just exists it is renamed to "s7prog.old". If your CPU is password protected so you have to add this as parameter. Otherwise there is no project written. Example: SAVE_PROJECT password	0xE806
FACTORY_RESET	Executes "factory reset".	0xE807
DIAGBUFF	The current diagnostics buffer of the CPU is stored as "diagbuff.txt" at the MMC.	0xE80B
SET_NETWORK	IP parameters for Ethernet PG/OP channel may be set by means of this command. The IP parameters are to be given in the order IP address, subnet mask and gateway in the format x.x.x.x each separated by a comma. Enter the IP address if there is no gateway used.	0xE80E
CMD_END	In the last line CMD_END is to be located.	0xE802

Examples

The structure of a command file is shown in the following. The corresponding diagnostics entry is put in parentheses.

Example 1

CMD_START	Marks the start of the command sequence (0xE801)
LOAD_PROJECT proj.wld	Execute an overall reset and load "proj.wld" (0xE805)
WAIT1SECOND	Wait ca. 1s (0xE803)
WEBPAGE	Store web page as "webpage.htm" (0xE804)
DIAGBUFF	Store diagnostics buffer of the CPU as "diagbuff.txt" (0xE80B)
CMD_END	Marks the end of the command sequence (0xE802)
... arbitrary text ...	Text after the command CMD_END is not evaluated.

Example 2

CMD_START	Marks the start of the command sequence (0xE801)
LOAD_PROJECT proj2.wld	Execute an overall reset and load "proj2.wld" (0xE805)
WAIT1SECOND	Wait ca. 1s (0xE803)
WAIT1SECOND	Wait ca. 1s (0xE803)
	IP parameter (0xE80E)
SET_NETWORK 172.16.129.210,255.255.224.0,172.16.129.210	
WAIT1SECOND	Wait ca. 1s (0xE803)
WAIT1SECOND	Wait ca. 1s (0xE803)
WEBPAGE	Store web page as "webpage.htm" (0xE804)
DIAGBUFF	Store diagnostics buffer of the CPU as "diagbuff.txt" (0xE80B)

VIPA specific diagnostic entries

CMD_END	Marks the end of the command sequence (0xE802)
... arbitrary text ...	Text after the command CMD_END is not evaluated.



The parameters IP address, subnet mask and gateway may be received from the system administrator.
Enter the IP address if there is no gateway used.

5.19 VIPA specific diagnostic entries

Entries in the diagnostic buffer

You may read the diagnostic buffer of the CPU via the Siemens SIMATIC Manager. Besides of the standard entries in the diagnostic buffer, the VIPA CPUs support some additional specific entries in form of event-IDs.

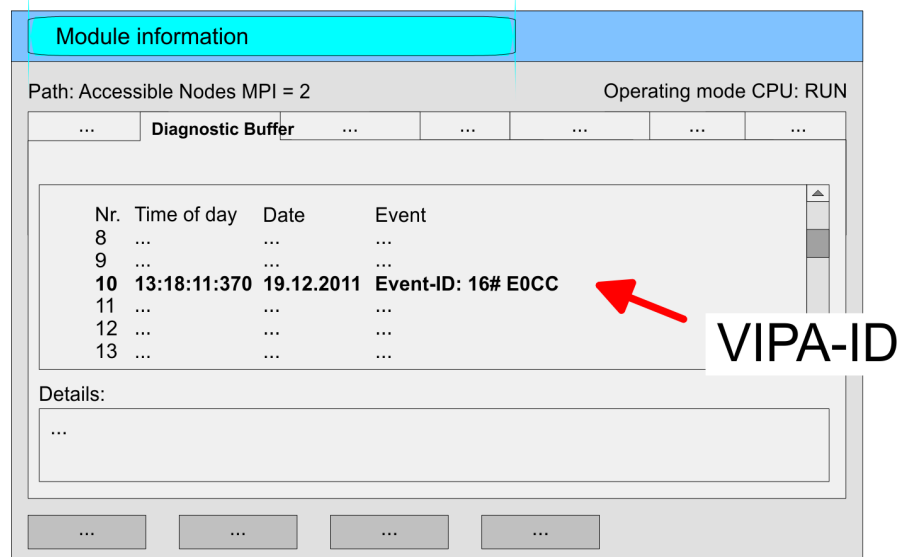
The current content of the diagnostics buffer is stored at the memory card by means of the CMD DIAGBUFF.



Every register of the module information is supported by the VIPA CPUs. More information may be found at the online help of the Siemens SIMATIC Manager.

Monitoring the diagnostic entries

To monitor the diagnostic entries you choose the option 'PLC → Module Information' in the Siemens SIMATIC Manager. Via the register "Diagnostic Buffer" you reach the diagnostic window:



The diagnosis is independent from the operating mode of the CPU. You may store a max. of 100 diagnostic entries in the CPU. The following page shows an overview of the VIPA specific Event-IDs.

Overview of the Event-IDs

Event-ID	Description
0x115C	Vendor-specific interrupt (OB 57) at EtherCAT OB: OB number (57) ZInfo1: Logical address of the slave, which has released the interrupt ZInfo2: Interrupt type ZInfo3: Reserved
0xE003	Error on accessing the periphery ZInfo1: Periphery address ZInfo2: Slot
0xE004	Multiple parametrization of a periphery address ZInfo1: Periphery address ZInfo2: Slot
0xE005	Internal error - Please contact the VIPA Hotline!
0xE006	Internal error - Please contact the VIPA Hotline!
0xE007	Configured in-/output bytes do not fit into periphery area
0xE008	Internal error - Please contact the VIPA Hotline!
0xE009	Error on accessing the standard backplane bus
0xE010	There is a undefined module at the backplane bus ZInfo2: Slot ZInfo3: Type ID
0xE011	Master project engineering at slave CPU not possible or wrong slave configuration
0xE012	Error at parametrization
0xE013	Error at shift register access to standard bus digital modules
0xE014	Error at Check_Sys
0xE015	Error at access to the master ZInfo2: Slot of the master (32=page frame master)
0xE016	Maximum block size at master transfer exceeded ZInfo1: Periphery address ZInfo2: Slot
0xE017	Error at access to integrated slave
0xE018	Error at mapping of the master periphery
0xE019	Error at standard back plane bus system recognition
0xE01A	Error at recognition of the operating mode (8 / 9 bit)

VIPA specific diagnostic entries

Event-ID	Description
0xE01B	Error - Maximum number of plug-in modules exceeded
0xE020	Error - Interrupt information is not defined
0xE030	Error of the standard bus
0xE033	Internal error - Please contact the VIPA Hotline!
0xE0B0	SPEED7 is not stoppable (Probably undefined BCD value at timer)
0xE0C0	Not enough space in work memory for storing code block (block size exceeded)
0xE0CB	Error at SSL access ZInfo1: 4=SSL wrong, 5=SubSSL wrong, 6=Index wrong ZInfo2: SSL-ID ZInfo3: Index
0xE0CC	Communication error MPI / Serial ZInfo1: Code 1: Wrong priority 2: Buffer overflow 3: Frame format error 4: Wrong SSL request (SSL-ID not valid) 5: Wrong SSL request (SSL-SubID not valid) 6: Wrong SSL request (SSL-Index not valid) 7: Wrong value 8: Wrong RetVal 9: Wrong SAP 10: Wrong connection type 11: Wrong sequence number 12: Faulty block number in the telegram 13: Faulty block type in the telegram 14: Inactive function 15: Wrong size in the telegram 20: Error writing to memory card 90: Faulty buffer size 98: Unknown error 99: Internal error
0xE0CD	Error at DP-V1 job management
0xE0CE	Error: Timeout at sending of the i-slave diagnostics
0xE0CF	Timeout at loading of a new HW configuration (timeout: 39 seconds)
0xE100	Memory card access error

Event-ID	Description
0xE101	Memory card error file system
0xE102	Memory card error FAT
0xE104	Memory card error at saving
0xE200	Memory card writing finished (Copy Ram2Rom)
0xE210	Memory card reading finished (reload after overall reset)
0xE21E	Memory card reading: Error at reload (after overall reset), file "Protect.wld" too big
0xE21F	Memory card reading: Error at reload (after overall reset), file read error, out of memory
0xE300	Internal flash writing finished (Copy Ram2Rom)
0xE310	Internal flash writing finished (reload after battery failure)
0xE311	Internal flash fx0000yy.wld file too big, load failure
0xE400	Memory card with the option memory expansion was plugged
0xE401	Memory card with the option memory expansion was removed
0xE402	The PROFIBUS DP master functionality is disabled. The interface acts further as MPI interface
0xE403	The PROFIBUS DP slave functionality is disabled. The interface acts further as MPI interface
0xE500	Memory management: Deleted block without corresponding entry in Block List ZInfo2: BlockType ZInfo3: BlockNo
0xE604	Multiple parametrization of a periphery address for Ethernet PG/OP channel ZInfo1: Periphery address ZInfo3: 0: Periphery address is input, 1: Periphery address is output
0xE701	Internal error - Please contact the VIPA Hotline!
0xE703	Internal error - Please contact the VIPA Hotline!
0xE720	Internal error - Please contact the VIPA Hotline!
0xE721	Internal error - Please contact the VIPA Hotline!
0xE801	CMD - Auto command: CMD_START recognized and successfully executed
0xE802	CMD - Auto command: CMD_End recognized and successfully executed
0xE803	CMD - Auto command: WAIT1SECOND recognized and successfully executed

VIPA specific diagnostic entries

Event-ID	Description
0xE804	CMD - Auto command: WEBPAGE recognized and successfully executed
0xE805	CMD - Auto command: LOAD_PROJECT recognized and successfully executed
0xE806	CMD - Auto command: SAVE_PROJECT ZInfo3: 0x0000: SAVE_PROJECT recognized and successfully executed ZInfo3: 0x8000: Error during SAVE_PROJECT e.g. wrong password
0xE807	CMD - Auto command: FACTORY_RESET recognized and successfully executed
0xE80B	CMD - Auto command: DIAGBUFF recognized and successfully executed
0xE80E	CMD - Auto command: SET_NETWORK recognized and successfully executed
0xE816	CMD - Auto command: SAVE_PROJECT: Error - CPU has been reset - no wld file was created.
0xE8FB	CMD - Auto command: Error: Initialization of the Ethernet PG/OP channel by means of SET_NETWORK is faulty
0xE8FC	CMD - Auto command: Error: Some IP parameters missing in SET_NETWORK
0xE8FE	CMD - Auto command: Error: CMD_START missing
0xE8FF	CMD - Auto command: Error: Error while reading CMD file (memory card error)
0xE901	Check sum error
0xEA00	Internal error - Please contact the VIPA Hotline!
0xEA01	Internal error - Please contact the VIPA Hotline!
0xEA02	SBUS: Internal error (internal plugged sub module not recognized) ZInfo1: Internal slot
0xEA03	SBUS: Communication error CPU - PROFINET I/O controller: ZInfo1: Slot ZInfo2: Status (0: OK, 1: ERROR, 2: BUSY, 3: TIMEOUT, 4: LOCKED, 5: UNKNOWN)
0xEA04	SBUS: Multiple parametrization of a periphery address ZInfo1: Periphery address ZInfo2: Slot ZInfo3: Data width
0xEA05	Internal error - Please contact the VIPA Hotline!
0xEA07	Internal error - Please contact the VIPA Hotline!

Event-ID	Description
0xEA08	SBUS: Parametrized input data width unequal to plugged input data width ZInfo1: Parametrized input data width ZInfo2: Slot ZInfo3: Input data width of the plugged module
0xEA09	SBUS: Parametrized output data width unequal to plugged output data width ZInfo1: Parametrized output data width ZInfo2: Slot ZInfo3: Output data width of the plugged module
0xEA10	SBUS: Input periphery address outside the periphery area ZInfo1: Periphery address ZInfo2: Slot ZInfo3: Data width
0xEA11	SBUS: Output periphery address outside the periphery area ZInfo1: Periphery address ZInfo2: Slot ZInfo3: Data width
0xEA12	SBUS: Error at writing record set ZInfo1: Slot ZInfo2: Record set number ZInfo3: Record set length
0xEA14	SBUS: Multiple parametrization of a periphery address (diagnostics address) ZInfo1: Periphery address ZInfo2: Slot ZInfo3: Data width
0xEA15	Internal error - Please contact the VIPA Hotline!
0xEA18	SBUS: Error at mapping of the master periphery ZInfo2: Slot of the master
0xEA19	Internal error - Please contact the VIPA Hotline!
0xEA20	Error - RS485 interface is not pre-set to PROFIBUS DP master bus a PROFIBUS DP master is configured.
0xEA21	Error - Configuration RS485 interface X2/X3: PROFIBUS DP master is configured but missing ZInfo2: Interface x
0xEA22	Error - RS485 interface X2 - Value exceeds the limits ZInfo: Configured value of X2
0xEA23	Error - RS485 interface X3 - Value exceeds the limits ZInfo: Configured value of X3

Event-ID	Description
0xEA24	Error - Configuration RS485 interface X2/X3: Interface/protocol missing, default settings are used ZInfo2: Configured value for X2 ZInfo3: Configured value for X3
0xEA30	Internal error - Please contact the VIPA Hotline!
0xEA40	Internal error - Please contact the VIPA Hotline!
0xEA41	Internal error - Please contact the VIPA Hotline!
0xEA50	Error - PROFINET configuration ZInfo1: User slot of the PROFINET I/O controller ZInfo2: IO-Device-No. ZInfo3: IO-Device slot
0xEA51	Error - There is no PROFINET IO controller at the configured slot ZInfo1: User slot of the PROFINET I/O controller ZInfo2: Recognized ID at the configured slot
0xEA53	Error - PROFINET configuration - There are too many PROFINET IO devices configured ZInfo1: Number of configured devices ZInfo2: Slot ZInfo3: Maximum possible number of devices
0xEA54	Error - PROFINET IO controller reports multiple parametrization of a periphery address ZInfo1: Periphery address ZInfo2: User slot of the PROFINET I/O controller ZInfo3: Data width
0xEA61 ... 0xEA63	Internal error - Please contact the VIPA Hotline!

Event-ID	Description
0xEA64	PROFINET/EtherCAT CP Configuration error: Zinfo1: Bit 0: Too many devices Bit 1: Too many devices per ms Bit 2: Too many input bytes per ms Bit 3: Too many output bytes per ms Bit 4: Too many input bytes per device Bit 5: Too many output bytes per device Bit 6: Too many productive connections Bit 7: Too many input bytes in the process image Bit 8: Too many output bytes in the process image Bit 9: Configuration not available Bit 10: Configuration not valid Bit 11: Cycle time too small Bit 12: Cycle time too big Bit 13: Not valid device number Bit 14: CPU is configured as I device Bit 15: Obtain an IP address in a different way is not supported for the IP address of the controller
0xEA65	Internal error - Please contact the VIPA Hotline!
0xEA66	PROFINET IO controller Error in communication stack PK: Rackslot OBNr: StackError.Service DatId: StackError.DeviceRef ZInfo1: StackError.Error.Code ZInfo2: StackError.Error.Detail ZInfo3: StackError.Error.AdditionalDetail << 8 + StackError.Error.AreaCode
0xEA67	Error - PROFINET IO controller - reading record set PK: Error type 0: DATA_RECORD_ERROR_LOCAL 1: DATA_RECORD_ERROR_STACK 2: DATA_RECORD_ERROR_REMOTE OBNr: PROFINET IO controller slot DatId: Device-No. ZInfo1: Record set number ZInfo2: Record set handle ZInfo3: Internal error code for service purposes

VIPA specific diagnostic entries

Event-ID	Description
0xEA68	Error - PROFINET IO controller - writing record set PK: Error type 0: DATA_RECORD_ERROR_LOCAL 1: DATA_RECORD_ERROR_STACK 2: DATA_RECORD_ERROR_REMOTE OBNo: PROFINET IO controller slot DatId: Device-No. ZInfo1: Record set number ZInfo2: Record set handle ZInfo3: Internal error code for service purposes
0xEA69	Internal error - Please contact the VIPA Hotline!
0xEA6A	PROFINET IO controller Service error in communication stack PK: Rackslot OBNo: ServiceIdentifier DatId: 0 ZInfo1: ServiceError.Code ZInfo2: ServiceError.Detail ZInfo3: StackError.Error.AdditionalDetail
0xEA6B	PROFINET IO controller Vendor ID mismatch PK: Rackslot OBNo: PLC Mode DatId: 0 ZInfo1: Device ID ZInfo2: - ZInfo3: -
0xEA6C	PROFINET IO controller Device ID mismatch PK: Rackslot OBNo: PLC Mode DatId: 0 ZInfo1: Device ID ZInfo2: - ZInfo3: -

Event-ID	Description
0xEA6D	PROFINET IO controller No empty name PK: Rackslot OBNo: PLC Mode DatId: 0 ZInfo1: Device ID ZInfo2: - ZInfo3: -
0xEA6E	PROFINET IO controller RPC response missing PK: Rackslot OBNo: PLC Mode DatId: 0 ZInfo1: Device ID ZInfo2: - ZInfo3: -
0xEA6F	PROFINET IO controller PN module mismatch PK: Rackslot OBNo: PLC-Mode DatId: 0 ZInfo1: Device ID ZInfo2: - ZInfo3: -
0xEA97	Storage error SBUS service channel ZInfo3 = Slot
0xEA98	Timeout at waiting for reboot of a SBUS module (server)
0xEA99	Error at file reading via SBUS

VIPA specific diagnostic entries

Event-ID	Description
0xEAA0	Emac Error occurred OBNo: Current PLC mode ZInfo1: Diagnostics address of the master / controller ZInfo2: 0: None Rx queue is full 1: No send buffer available 2: Send stream was cut off; sending failed 3: Exhausted retries 4: No receive buffer available in Emac DMA 5: Emac DMA transfer aborted 6: Queue overflow 7: Unexpected frame received ZInfo3: Number of errors, which occurred
0xEAB0	Link mode not valid OBNo: Current PLC mode ZInfo1: Diagnostics address master / controller Zinfo2: Current LinkMode 0x01: 10Mbit full-duplex 0x02: 100Mbit half-duplex 0x03: 100Mbit full-duplex 0x05: 10Mbit half-duplex 0xFF: Link mode not defined
0xEB03	SLIO error on IO mapping
0xEB10	SLIO error: Bus error ZInfo1: Type of error 0x82: ErrorAlarm
0xEB20	SLIO error: Interrupt information undefined
0xEB21	SLIO error on accessing the configuration data

Event-ID	Description
0xEC03	<p>EtherCAT: Configuration error</p> <p>ZInfo1: Errorcode</p> <p>1: NUMBER_OF_SLAVES_NOT_SUPPORTED</p> <p>2: SYSTEM_IO_NR_INVALID</p> <p>3: INDEX_FROM_SLOT_ERROR</p> <p>4: MASTER_CONFIG_INVALID</p> <p>5: MASTER_TYPE_ERROR</p> <p>6: SLAVE_DIAG_ADDR_INVALID</p> <p>7: SLAVE_ADDR_INVALID</p> <p>8: SLAVE_MODULE_IO_CONFIG_INVALID</p> <p>9: LOG_ADDR_ALREADY_IN_USE</p> <p>10: NULL_PTR_CHECK_ERROR</p> <p>11: IO_MAPPING_ERROR</p> <p>12: ERROR</p>
0xEC04	<p>EtherCAT: Multiple configuration of a periphery address</p> <p>ZInfo1: Periphery address</p> <p>ZInfo2: Slot</p>
0xEC10	<p>EtherCAT: Restoration bus with its slaves</p> <p>OB start Info (Local data) StartEvent and Eventclass: 0xEC10</p> <p>DatID:</p> <p>0xXXYY:</p> <p>XX=0x54 with input address in ZInfo1,</p> <p>XX=0x55 with output address.</p> <p>YY=0x00 Station not available,</p> <p>YY=0x01 Station available (process data)</p> <p>ZInfo1: 0xXXYY (XX=OldState, YY=NewState)</p> <p>ZInfo2: Diagnostics address of the master</p> <p>ZInfo3: Number of stations, which are not in the same state as the master (> 0)</p>

Event-ID	Description
0xEC11	<p>EtherCAT: Restoration bus with missing slaves OB start Info (Local data) StartEvent and Eventclass: 0xEC11 DatID: 0xXXYY: XX=0x54 with input address in ZInfo1, XX=0x55 with output address. YY=0x00 Station not available, YY=0x01 Station available (process data) ZInfo1: 0xXXYY (XX=OldState, YY=NewState) ZInfo2: Diagnostics address of the master ZInfo3: Number of stations, which are not in the same state as the master (> 0)</p>
0xEC12	<p>EtherCAT: restoration slave OB start Info (Local data) StartEvent and Eventclass: 0xEC12 DatID: 0xXXYY: XX=0x54 with input address in ZInfo1, XX=0x55 with output address. YY=0x00 Station not available, YY=0x01 Station available (process data) ZInfo1: 0xXXYY (XX=OldState, YY=NewState) ZInfo2: Diagnostics of the Station ZInfo3: AIStatusCode</p>
0xEC30	<p>EtherCAT: Topology OK OB start Info (Local data) StartEvent and Eventclass: 0xEC30 ZInfo2: Diagnostics address of the master</p>
0xEC50	<p>EtherCAT: DC not in Sync ZInfo1: Diagnostics address of the master</p>
0xED10	<p>EtherCAT: Bus failure OB start Info (Local data) StartEvent and Eventclass: 0xED10 DatID: 0xXXYY: XX=0x54 with input address in ZInfo1, XX=0x55 with output address. YY=0x00 Station not available, YY=0x01 Station available (process data) ZInfo1: 0xXXYY (XX=OldState, YY=NewState) ZInfo2: Diagnostics address of the master ZInfo3: Number of stations, which are not in the same state as the master</p>

Event-ID	Description
0xED12	<p>EtherCAT: Failure slave</p> <p>OB start Info (Local data) StartEvent and Eventclass: 0xED12</p> <p>DatID:</p> <p>0xXXYY:</p> <p>XX=0x54 with input address in ZInfo1, XX=0x55 with output address.</p> <p>YY=0x00 Station not available, YY=0x01 Station available (process data)</p> <p>ZInfo1: 0xXXYY (XX=OldState, YY=NewState)</p> <p>ZInfo2: Diagnostics of the Station</p> <p>ZInfo3: AIStatusCode</p>
0xED20	<p>EtherCAT: Bus state change without calling OB86</p> <p>OB start Info (Local data) StartEvent and Eventclass: 0xED20</p> <p>DatID:</p> <p>0xXXYY:</p> <p>XX=0x54 with input address in ZInfo1, XX=0x55 with output address.</p> <p>YY=0x00 Station not available, YY=0x01 Station available (process data)</p> <p>ZInfo1: 0xXXYY (XX=OldState, YY=NewState)</p> <p>ZInfo2: Diagnostics address of the master</p> <p>ZInfo3: Number of stations, which are not in the same state as the master</p>
0xED21	<p>EtherCAT: error in bus state change</p> <p>OB: 0x00</p> <p>PK: 0x00</p> <p>DatID:</p> <p>0xXXYY:</p> <p>XX=0x54 with input address in ZInfo1, XX=0x55 with output address.</p> <p>YY=0x00 Station not available, YY=0x01 Station available (process data)</p> <p>ZInfo1: 0xXXYY (XX = current state, YY = expected state)</p> <p>ZInfo2: Diagnostics address of the master</p> <p>ZInfo3: ErrorCode:</p> <p>0x0008: Busy</p> <p>0x000B: Invalid Parameter</p> <p>0x000E: Invalid State</p> <p>0x0010: Timeout</p>

VIP A specific diagnostic entries

Event-ID	Description
0xED22	<p>EtherCAT: Bus state change without calling OB86</p> <p>OB start Info (Local data) StartEvent and Eventclass: 0xED22</p> <p>DatID:</p> <p>0xXXYY:</p> <p>XX=0x54 with input address in ZInfo1, XX=0x55 with output address.</p> <p>YY=0x00 Station not available, YY=0x01 Station available (process data)</p> <p>ZInfo1: 0xXXYY (XX=OldState, YY=NewState)</p> <p>ZInfo2: Diagnostics of the Station</p> <p>ZInfo3: AIStatusCode</p>
0xED30	<p>EtherCAT: Topology Mismatch</p> <p>OB start Info (Local data) StartEvent and Eventclass: 0xED30</p> <p>ZInfo2: Diagnostics address of the master</p>
0xED31	<p>EtherCAT: Interrupt Queue Overflow</p> <p>OB start Info (Local data) StartEvent and Eventclass: 0xED31</p> <p>ZInfo2: Diagnostics address of the master</p>
0xED40 ... 0xED4F	Internal error - Please contact the VIP A Hotline!
0xED50	<p>EtherCAT: DC not in Sync</p> <p>ZInfo1: Diagnostics address of the master</p>
0xED60	<p>EtherCAT: Diagnostics buffer CP:</p> <p>Slave state change</p> <p>PK: 0</p> <p>OB: PLC-Mode</p> <p>DatID 1/2: 0</p> <p>ZInfo1: 0x00YY:</p> <p>YY: New EtherCAT state of the slave</p> <p>ZInfo2: EtherCAT station address</p> <p>ZInfo3: AIStatusCode (EtherCAT specific error code)</p>

Event-ID	Description
0xED61	<p>EtherCAT: Diagnostics buffer CP: CoE emergency PK: EtherCAT station address (low byte) OB: EtherCAT station address (high byte) DatID 1/2: Error code ZInfo1: 0xYYZZ: YY: Error register ZZ: MEF byte 1 ZInfo 2: 0xYYZZ: YY: MEF byte 2 ZZ: MEF byte 3 ZInfo3: 0xYYZZ: YY: MEF byte 4 ZZ: MEF byte 5</p>
0xED62	<p>EtherCAT: Diagnostics buffer CP: Error on SDO access during state change PK: EtherCAT station address (low byte) OB: EtherCAT station address (high byte) DatID 1/2: Subindex ZInfo1: Index ZInfo2: SDO error code (high word) ZInfo3: SDO error code (low word)</p>
0xED70	<p>EtherCAT: Diagnostics buffer CP: Twice HotConnect group found PK: 0 OB: PLC-Mode DatID 1/2: 0 ZInfo1: Diagnostics address of the master ZInfo2: EtherCAT station address ZInfo3: 0</p>
0xEE00	Additional information at UNDEF_OPCODE
0xEE01	Internal error - Please contact the VIP A Hotline!
0xEEEE	CPU was completely overall reset, since after PowerON the start-up could not be finished.
0xEF11 ... 0xEF13	Internal error - Please contact the VIP A Hotline!

Event-ID	Description
0xEFFF	Internal error - Please contact the VIPA Hotline!
PK: C-Source module number DatID: Line number	

5.20 Control and monitoring of variables with test functions

Overview

For troubleshooting purposes and to display the status of certain variables you can access certain test functions via the menu item **Debug** of the Siemens SIMATIC Manager.

- The status of the operands and the RLO can be displayed by means of the test function *'Debug → Monitor'*.
- The status of the operands and the RLO can be displayed by means of the test function *'PLC → Monitor/Modify Variables'*.

'Debug → Monitor'

This test function displays the current status and the RLO of the different operands while the program is being executed. It is also possible to enter corrections to the program.



When using the test function "Monitor" the PLC must be in RUN mode!

The processing of the states may be interrupted by means of jump commands or by timer and process-related interrupts. The interruption of the processing of statuses does not change the execution of the program. It only shows that the data displayed is no longer valid. At the breakpoint the CPU stops collecting data for the status display and instead of the required data it only provides the PG with data containing the value 0. For this reason, jumps or time and process alarms can result in the value displayed during program execution remaining at 0 for the items below:

- the result of the logical operation RLO
- Status / AKKU 1
- AKKU 2
- Condition byte
- absolute memory address SAZ. In this case SAZ is followed by a "?".

'PLC → Monitor/Modify Variables'

This test function returns the condition of a selected operand (inputs, outputs, flags, data word, counters or timers) at the end of program execution. This information is obtained from the process image of the selected operands. During the "processing check" or in operating mode STOP the periphery is read directly from the inputs. Otherwise only the process image of the selected operands is displayed.

- Control of outputs
 - It is possible to check the wiring and proper operation of output modules.
 - You can set outputs to any desired status with or without a control program. The process image is not modified but outputs are no longer inhibited.
- Control of variables
 - The following variables may be modified: I, Q, M, T, C and D.
 - The process image of binary and digital operands is modified independently of the operating mode of the CPU.
 - When the operating mode is RUN the program is executed with the modified process variable. When the program continues they may, however, be modified again without notification.
 - Process variables are controlled asynchronously to the execution sequence of the program.

6 Deployment I/O periphery

6.1 Overview

Hardware

At the 312-5BE13 the connectors for digital in-/output and technological functions are integrated to a 2tier casing.

Project engineering

The project engineering takes place in the Siemens SIMATIC manager as CPU 312C from Siemens (6ES7 312-5BE03-0AB0 V2.6). Here the CPU 312-5BE13 is parameterized by the "Properties" dialog of the CPU 312C. For parameterization of the digital I/O periphery and the technological functions the corresponding submodule of the CPU 312C may be used.

I/O periphery

The integrated I/Os of the 312-5BE13 may be used for technological functions or as standard I/Os. Technological functions and standard I/Os may be used simultaneously with appropriate hardware. Read access to inputs used by technological functions is possible. Write access to used outputs is not possible.

Technological functions

Up to 2 channels may be parameterized as technological function. The parameterization of the appropriate channel is made in the hardware configurator by the *count* submodule of the CPU 312C. There are the following technological functions:

- Continuous count
- Single count
- Periodic count
- Frequency measurement
- Pulse width modulation (PWM)

The controlling of the corresponding counter mode happens by means of the SFB COUNT (SFB 47) of the user program.

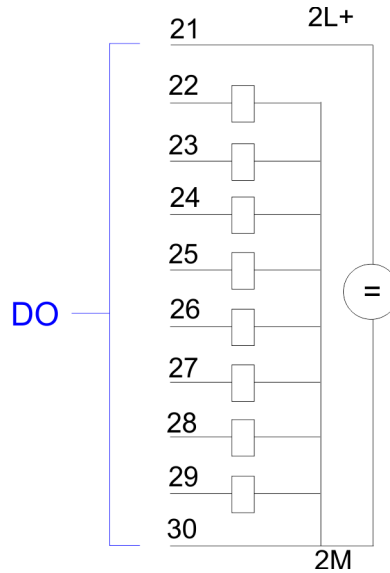
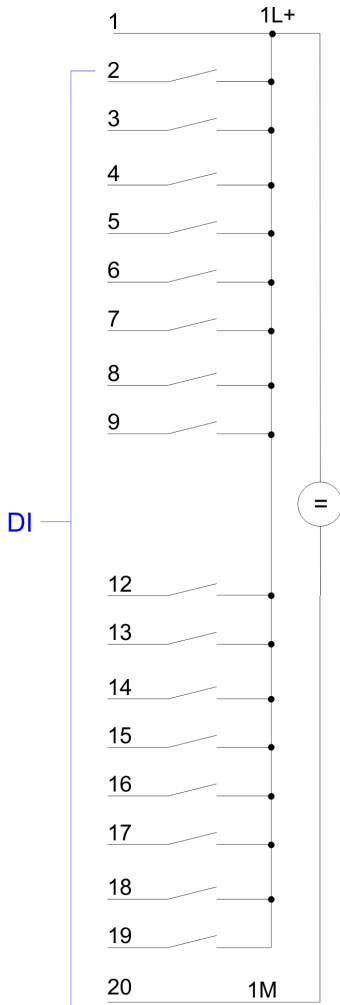
6.2 In-/Output area CPU 312-5BE13


Overview

The CPU 312-5BE13 has the following digital in and output channels integrated in one casing:

- Digital Input: 16xDC 24V, with interrupt capability
- Digital Output: 8xDC 24V, 0.5A
- Technological functions: 2 channels

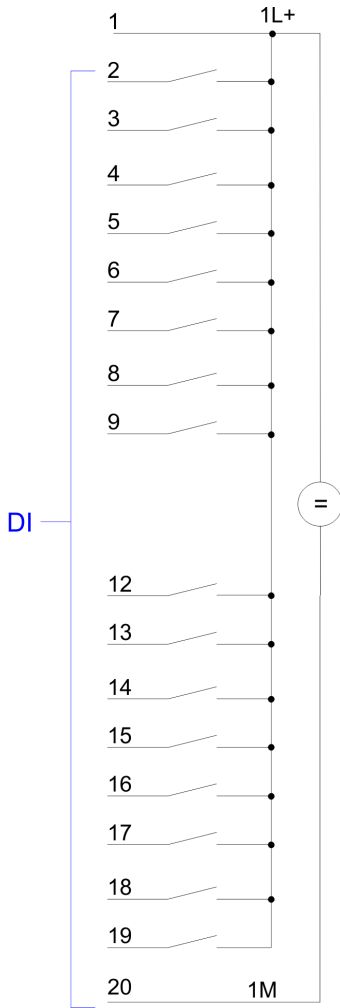
X11:





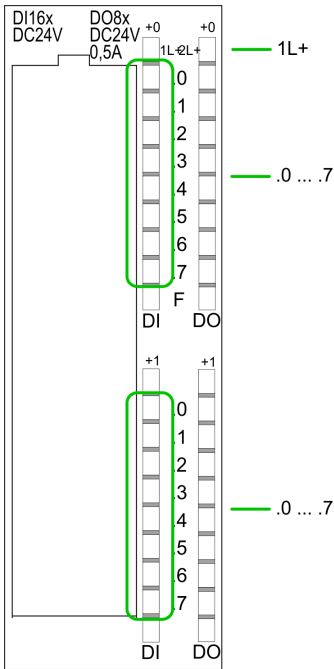
CAUTION! Please regard that the voltage at an output channel is always \leq the supply voltage connected to L+.

In-/Output area CPU 312-5BE13



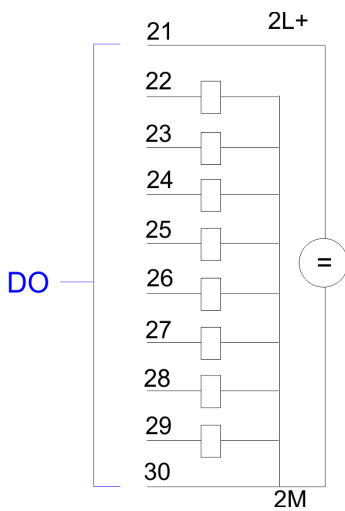
Pin assignment X11: DI

Pin	Assignment
1	1L+ Power supply +DC 24V
2	I+0.0 / Channel 0 (A) / Pulse
3	I+0.1 / Channel 0 (B) / Direction
4	I+0.2 / Channel 0 HW gate
5	I+0.3 / Channel 1 (A) / Pulse
6	I+0.4 / Channel 1 (B) / Direction
7	I+0.5 / Channel 1 HW gate
8	I+0.6
9	I+0.7
10	not used
11	not used
12	I+1.0
13	I+1.1
14	I+1.2
15	I+1.3
16	I+1.4 / Channel 0 Latch
17	I+1.5 / Channel 1 Latch
18	I+1.6
19	I+1.7
20	Ground 1M DI



Status indication X11: DI

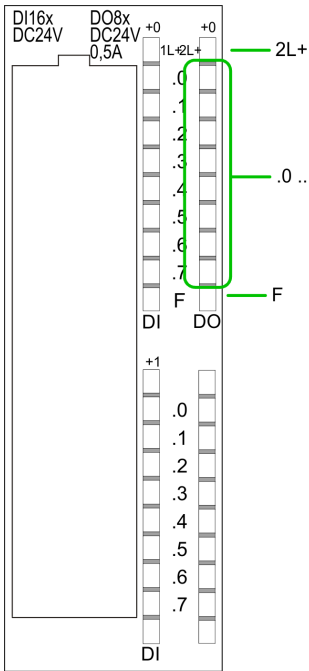
- 1L+
 - LED (green)
 - Supply voltage available for DI
- .07
 - LEDs (green)
 - I+0.0 ... I+0.7
 - I+1.0 ... I+1.7
 - Starting with ca. 15V the signal "1" at the input is recognized and the according LED is activated



Pin assignment X11: DO

Pin	Assignment
21	2L+ Power supply +DC 24V
22	O+0.0 / Channel 0 Output
23	O+0.1 / Channel 1 Output
24	Q+0.2
25	Q+0.3
26	Q+0.4
27	Q+0.5
28	Q+0.6
29	Q+0.7
30	Ground 2M DO
31 ... 40	not used

Address assignment



Status indication X11: DO

- 2L+
 - LED (green)
 - Supply voltage available for DO
- .07
 - LEDs (green)
 - Q+0.0 ... Q+0.7
 - The according LED is on at active output
- F
 - LED (red)
 - Overload or short circuit error

6.3 Address assignment

Input range

Sub module	Default address	Access	Assignment
DI10/DO6	124	Byte	Digital Input I+0.0 ... I+0.7
	125	Byte	Digital Input I+1.0 ... I+1.7
Counter	768	DInt	Channel 0: Count value / Frequency value
	772	DInt	Channel 1: Count value / Frequency value
	776	DInt	reserved
	780	DInt	reserved

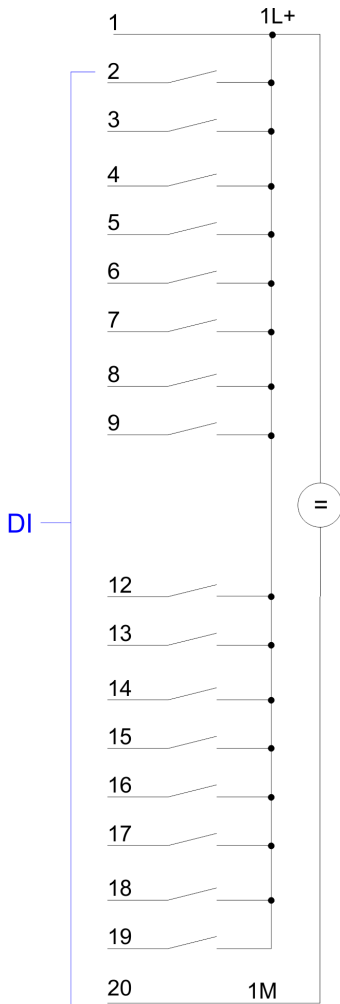
Output range

Sub module	Default address	Access	Assignment
DI10/DO6	124	Byte	Digital Output Q+0.0 ... Q+0.7
Counter	768	DWort	reserved
	772	DWort	reserved
	776	DWort	reserved
	780	DWort	reserved

6.4 Digital part

312-5BE13

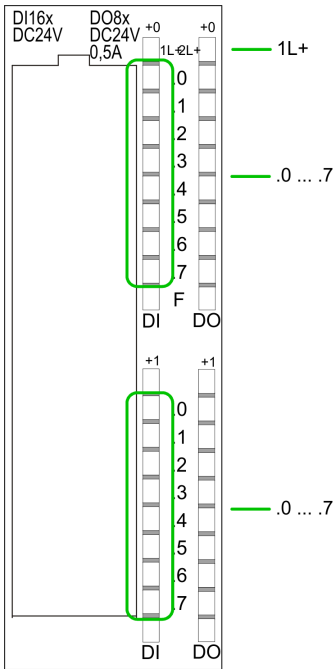
The digital part consists of 16 input -, 8 output channels and 2 channels for technological functions. Each of these digital input- respectively output channels show its state via a LED. By means of the parameterization you may assign interrupt properties to the inputs I+0.0 to I+1.1.



Pin assignment X11: DI

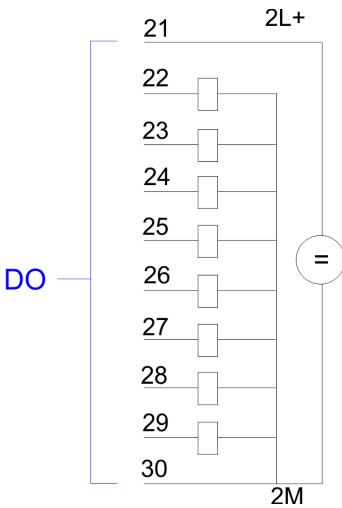
Pin	Assignment
1	1L+ Power supply +DC 24V
2	I+0.0 / Channel 0 (A) / Pulse
3	I+0.1 / Channel 0 (B) / Direction
4	I+0.2 / Channel 0 HW gate
5	I+0.3 / Channel 1 (A) / Pulse
6	I+0.4 / Channel 1 (B) / Direction
7	I+0.5 / Channel 1 HW gate
8	I+0.6
9	I+0.7
10	not used
11	not used
12	I+1.0
13	I+1.1
14	I+1.2
15	I+1.3
16	I+1.4 / Channel 0 Latch
17	I+1.5 / Channel 1 Latch
18	I+1.6
19	I+1.7
20	Ground 1M DI

Digital part



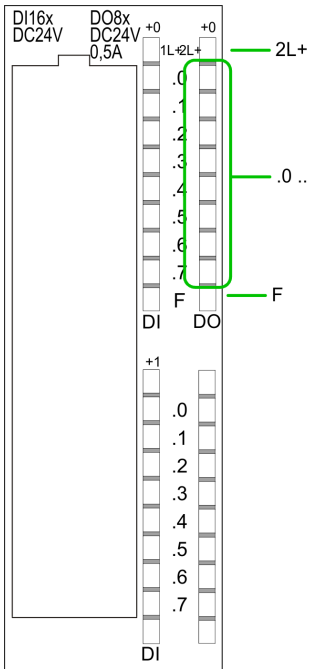
Status indication X11: DI

- 1L+
 - LED (green)
 - Supply voltage available for DI
- .07
 - LEDs (green)
 - I+0.0 ... I+0.7
 - I+1.0 ... I+1.7
 - Starting with ca. 15V the signal "1" at the input is recognized and the according LED is activated



Pin assignment X11: DO

Pin	Assignment
21	2L+ Power supply +DC 24V
22	O+0.0 / Channel 0 Output
23	O+0.1 / Channel 1 Output
24	Q+0.2
25	Q+0.3
26	Q+0.4
27	Q+0.5
28	Q+0.6
29	Q+0.7
30	Ground 2M DO
31 ... 40	not used



Status indication X11: DO

- 2L+
 - LED (green)
 - Supply voltage available for DO
- .07
 - LEDs (green)
 - Q+0.0 ... Q+0.7
 - The according LED is on at active output
- F
 - LED (red)
 - Overload or short circuit error

6.4.1 Access to the I/O area

The 312-5BE13 creates in its peripheral area an area for input respectively output data. Without a hardware configuration the in the following specified default addresses are used.

6.4.1.1 Address assignment

Input range

Sub module	Default address	Access	Assignment
DI10/DO6	124	Byte	Digital Input I+0.0 ... I+0.7
	125	Byte	Digital Input I+1.0 ... I+1.7
Counter	768	DInt	Channel 0: Count value / Frequency value
	772	DInt	Channel 1: Count value / Frequency value
	776	DInt	reserved
	780	DInt	reserved

Output range

Sub module	Default address	Access	Assignment
DI10/DO6	124	Byte	Digital Output Q+0.0 ... Q+0.7
Counter	768	DWort	reserved
	772	DWort	reserved

Sub module	Default address	Access	Assignment
	776	DWort	reserved
	780	DWort	reserved

6.4.2 Parameterization - Digital part

Parameter data	Parameters of the digital part may be set by means of the <i>DI10/DO6</i> submodule of the CPU 312C from Siemens during hardware configuration. In the following all parameters are specified, which may be used with the hardware configuration of the digital periphery.
General	This provides the short description of the digital periphery. At <i>Comment</i> information about the module such as purpose may be entered.
Addresses	At this register the start address of the in-/output periphery may be set.
Inputs	Here there are the following adjustment possibilities: <ul style="list-style-type: none"> ■ Hardware interrupt <ul style="list-style-type: none"> – A hardware interrupt may be optionally triggered on the rising or falling edge of an input. A diagnostic interrupt is only supported together with hardware interrupt lost. Select with the arrow keys the input and activate the desired hardware interrupt. ■ Input delay <ul style="list-style-type: none"> – The input delay may be configured per channel in groups of four. Please note that in the parameter window only the value 0.1ms may be set. At the other values 0.35ms is internally used for input delay.
Outputs	There are no parameters for the digital output channels.

6.5 Counter

6.5.1 Counter - Fast introduction

Overview	<p>The CPU 312-5BE13 has in-/outputs, which may be used for technological functions respectively as standard periphery. Technological functions and standard I/O may be used simultaneously with appropriate hardware. Read access to inputs used by technological functions is possible. Write access to used outputs is not possible. The parameterization of the corresponding channel is made in the hardware configurator by means of the <i>Count</i> submodule of the CPU 312C from Siemens. Now the following technological functions at 2 channels are at the disposal:</p> <ul style="list-style-type: none"> ■ Continuous count, e.g. for position decoding with Incremental encoder ■ Single count, e.g. for unit decoding to a maximum limit ■ Periodical count, e.g. for applications with repeated counting operations
-----------------	--

Independent of the number of activated counters for the CPU 312-5BE13 the maximum frequency amounts to 10kHz.

The controlling of the appropriate modes of operation is made from the user program by the SFB COUNT (SFB 47).

Pin assignment

↪ Chapter 6.4 'Digital part' on page 95

Preset respectively parameterize counter

The counter signal is detected and evaluated during counting operation. Every counter occupies one double word in the input range for the *counter register*. In the operating modes "single count" and "periodical count" an end respectively start value may be defined according to the counting direction up respectively down. Each counter has parameterizable additional functions as gate function, latch function, comparison value, hysteresis and hardware interrupt. Each counter parameter may be set by the *Count* submodule of the Siemens CPU 312C. Here is defined among others:

- Interrupt behavior
- Max. frequency
- Counter mode respectively behaviour
- Stat, end, comparison value and hysteresis

Parameterization

1. ▶ Start the Siemens SIMATIC Manager with your project and open the hardware configurator.
2. ▶ Place a profile rail.
3. ▶ Configure at slot 2 the corresponding CPU from Siemens CPU 312C.
4. ▶ Open the dialog window "Properties" by a double click to the *Count* submodule of the CPU.
5. ▶ As soon as an operating mode to the corresponding channel is selected, a dialog window for this operating mode is created and displayed and filled with default parameters.
6. ▶ Execute the wished parameterization.
7. ▶ Save the project with '*Station → Save and compile*'.
8. ▶ Transfer the project to the CPU.

Controlling the counter functions

The SFB COUNT (SFB 47) should cyclically be called (e.g. OB 1) for controlling the counter functions. The SFB is to be called with the corresponding instance DB. Here the parameters of the SFB are stored. Among others the SFB 47 contains a request interface. Hereby you get read and write access to the registers of the appropriate counter. So that a new job may be executed, the previous job must have been finished with `JOB_DONE = TRUE`. Per channel you may call the SFB in each case with the same instance DB, since the data necessary for the internal operational are stored here. Writing accesses to outputs of the instance DB is not permissible.



You must not call an SFB you have configured in your program in another program section under another priority class, because the SFB must not interrupt itself. Example: It is not allowed to call the same SFB both in OB 1 and in the interrupt OB.

Controlling the counter The counter is controlled by the internal gate (i gate). The i gate is the result of logic operation of hardware gate (HW gate) and software gate (SW gate), where the HW gate evaluation may be deactivated by the parameterization.

HW gate: open (activate):	Edge 0-1 at hardware gate _x input of the module
close (deactivate):	Edge 1-0 at hardware gate _x input of the module
SW gate: open (activate):	In application program by setting SW_GATE of the SFB 47
close (deactivate):	In application program by resetting SW_GATE of the SFB 47

Read counter The counter values may be read by the output parameter COUNTVAL of the SFB 47. There is also the possibility for direct access to the counter values by means of the input address of the *Count* submodule.

Counter inputs (Connections)

There are the following possibilities for connection to the technological functions:

- 24V incremental encoder, equipped with two tracks with 90° phase offset
- 24V pulse generator with direction signal
- 24V proximity switch (e.g. BERO or light barrier)

For not all inputs are available at the same time, you may set the input assignment for every counter via the parameterization. For each counter the following inputs are available:

- *Channel_x (A)*
Pulse input for count signal res. track A of an encoder. Here you may connect encoder with 1-, 2- or 4-tier evaluation.
- *Channel_x (B)*
Direction signal res. track B of the encoder. Via the parameterization you may invert the direction signal.
- *Hardware gate_x*
This input allows you to open the HW gate with a high peak and thus start a count process. The usage of the HW gate may be parameterized.
- *Latch_x*
With an edge 0-1 at Latch_x the recent counter value is stored in a memory that you may read at need.

Counter outputs

Every counter has an assigned output channel. The following behavior for the output channel may be set via parameterization:

- No comparison: Output is not controlled and is switched in the same way as a normal output.
- Count value \geq comparison value: Output is set as long as counter value \geq comparison value.

- Count value \leq comparison value: Output is set as long as counter value \leq comparison value.
- Pulse at comparison value: You can specify a pulse period for adaptation to the actuators you are using. The output is set for the given pulse duration, as soon as the counter reached the comparison value. If you have parameterized a main count direction the output is only set when reaching the comparison value from the main counting direction. The maximum pulse duration may amount to 510ms. By setting 0 as pulse duration the output gets set as long as the comparison conditions are fulfilled.

6.5.1.1 Parameter overview

In the following the parameters are listed which may be used for counter configuration during hardware configuration.

General

Here the short description of the counter function may be found. At *Comment* information about the module such as purpose may be entered.

Addresses

Here the start address of the in- output periphery is set.

Basic parameters

Here the interrupts the counter functions should trigger may be selected. You have the following options:

- None: There is no interrupt triggered.
- Process: The counting function triggers a hardware interrupt.
- Diagnostics and Process: With the 312-5BE13 the diagnostic interrupt of the digital in-/output periphery is only supported in connection with "hardware interrupt lost".

Count

Parameters	Description	Range of values	Default
Main count direction	<ul style="list-style-type: none"> ■ <i>None</i>: No restriction of the counting range ■ <i>Up</i>: Restricts the up-counting range. Counter starts at 0 or load value, counts in positive direction up to the declaration end value -1 and then jumps back to load value at the next positive transducer pulse. ■ <i>Down</i>: Restricts the down-counting range. The Counter starts at the declared start value or load value in negative direction, counts to 1 and then jumps to start value at the next negative encoder pulse. 	<ul style="list-style-type: none"> ■ None ■ Up ■ Down (not with continuous count) 	<ul style="list-style-type: none"> ■ None
End value/ Start value	<p><i>End value</i>, with up-count as default.</p> <p><i>Start value</i>, with down-count as default.</p>	2...2147483647 ($2^{31}-1$)	2147483647 ($2^{31}-1$)

Parameters	Description	Range of values	Default
Gate function	<ul style="list-style-type: none"> ■ <i>Cancel count</i>: The count starts when the gate opens and resumes at the load value when the gate opens again. ■ <i>Stop count</i>: The count is interrupted when the gate closes and resumed at the last actual value when the gate opens again. 	<ul style="list-style-type: none"> ■ Abort the count operation ■ Interrupt the count operation 	Cancel count
Comparison value	<p>The count value is compared with the comparison value. see also the parameter "Characteristics of the output":</p> <ul style="list-style-type: none"> ■ No main direction of count ■ Up-count as default ■ Down-count as default 	<p>-2^{31} to $+2^{31}-1$</p> <p>-2^{31} to End value</p> <p>1 to $+2^{31}-1$</p>	0
Hysteresis	<p>A hysteresis is used to eliminate frequent output jitter if the count value lies within the range of the comparison value.</p> <p>0 and 1 means: Hysteresis switched off</p>	0 to 255	0
max. frequency: counting signals/ hardware gate	You can set the maximum frequency of the track A/pulse, track B/direction and hardware gate signals in fixed steps.	10, 5, 2, 1kHz	10kHz
max. frequency: Latch	You can set the maximum frequency of the latch signal in fixed steps.	10, 5, 2, 1kHz	10kHz
Signal evaluation	<p>The count and direction signals are connected to the input.</p> <p>A rotary transducer is connected to the input (single, dual or quadruple evaluation).</p>	<ul style="list-style-type: none"> ■ Pulse/Direction ■ Rotary encoder single ■ Rotary encoder, double ■ Rotary encoder quadruple 	Pulse/Direction
Hardware gate	In the activated state the Gate control is made via SWgate and HW-gate, otherwise via SW-gate only.	<ul style="list-style-type: none"> ■ activated ■ deactivated 	deactivated
Count direction inverted	In the activated state the "direction" input signal is inverted.	<ul style="list-style-type: none"> ■ activated ■ deactivated 	deactivated
Characteristics of the output	The output and the "Comparator" (STS_CMP) status bit are set, dependent on this parameter.	<ul style="list-style-type: none"> ■ No comparison ■ $\text{Count} \geq \text{comparison value}$ ■ $\text{Count} \leq \text{comparison value}$ ■ Pulse at comparison value 	No comparison

Parameters	Description	Range of values	Default
Pulse duration	With the setting "Characteristics of the output: Pulse at comparison value" the pulse duration of the output signal may be specified. Only even values are possible. The value is internal multiplied with 1.024ms.	0 to 510	0
Hardware interrupt: Hardware gate opening	In the activated state a hardware interrupt is generated when the hardware gate opens while the software gate is open.	<input type="checkbox"/> activated <input type="checkbox"/> deactivated	
Hardware interrupt: Hardware gate closing	In the activated state a hardware interrupt is generated when the hardware gate closes while the software gate is open.	<input type="checkbox"/> activated <input type="checkbox"/> deactivated	deactivated
Hardware interrupt: On reaching comparator	In the activated state a hardware interrupt is triggered on reaching the comparator (reaction) value. The process interrupt may only be released if in addition the value of "Characteristics of the output" is not "no comparison".	<input type="checkbox"/> activated <input type="checkbox"/> deactivated	deactivated
Hardware interrupt: Overflow	In the activated state a hardware interrupt is generated in the event of an overflow (exceeding the upper count limit).	<input type="checkbox"/> activated <input type="checkbox"/> deactivated	deactivated
Hardware interrupt: Underflow	In the activated state a hardware interrupt is generated in the event of an underflow (undershooting the lower count limit).	<input type="checkbox"/> activated <input type="checkbox"/> deactivated	deactivated

6.5.2 SFB 47 - COUNT - Counter controlling

Description

The SFB 47 is a specially developed block for the VIPA CPU for controlling of the counters. The SFB is to be called with the corresponding instance DB. Here the parameters of the SFB are stored. With the SFB COUNT (SFB 47) you have following functional options:

- Start/Stop the counter via software gate *SW_GATE*
- Enable/control digital output DO
- Read the status bit
- Read the actual count and latch value
- Request to read/write internal counter registers

Parameters

Name	Data type	Address (Instance DB)	Default value	Comment
LADDR	WORD	0.0	300h	This parameter is not evaluated. Always the internal I/O periphery is addressed.
CHANNEL	INT	2.0	0	Channel number
SW_GATE	BOOL	4.0	FALSE	Enables the Software gate
CTRL_DO	BOOL	4.1	FALSE	Enables the output False: Standard Digital Output
SET_DO	BOOL	4.2	FALSE	Parameter is not evaluated
JOB_REQ	BOOL	4.3	FALSE	Initiates the job (edge 0-1)
JOB_ID	WORD	6.0	0	Job ID
JOB_VAL	DINT	8.0	0	Value for write jobs
STS_GATE	BOOL	12.0	FALSE	Status of the internal gate
STS_STRT	BOOL	12.1	FALSE	Status of the hardware gate
STS_LTCH	BOOL	12.2	FALSE	Status of the latch input
STS_DO	BOOL	12.3	FALSE	Status of the output
STS_C_DN	BOOL	12.4	FALSE	Status of the down-count Always indicates the last direction of count. After the first SFB call <i>STS_C_DN</i> is set FALSE.
STS_C_UP	BOOL	12.5	FALSE	Status of the up-count Always indicates the last direction of count. After the first SFB call <i>STS_C_UP</i> is set TRUE.
COUNTVAL	DINT	14.0	0	Actual count value
LATCHVAL	DINT	18.0	0	Actual latch value
JOB_DONE	BOOL	22.0	TRUE	New job can be started
JOB_ERR	BOOL	22.1	FALSE	Job error
JOB_STAT	WORD	24.0	0	Job error ID

Local data only in instance DB

Name	Data type	Address (Instance DB)	Default value	Comment
RES00	BOOL	26.0	FALSE	reserved
RES01	BOOL	26.1	FALSE	reserved
RES02	BOOL	26.2	FALSE	reserved
STS_CMP	BOOL	26.3	FALSE	Comparator Status * Status bit <i>STS_CMP</i> indicates that the comparison condition of the comparator is or was reached. <i>STS_CMP</i> also indicates that the output was set. (<i>STS_DO</i> = TRUE).
RES04	BOOL	26.4	FALSE	reserved
STS_OFLW	BOOL	26.5	FALSE	Overflow status *
STS_UFLW	BOOL	26.6	FALSE	Underflow status *
STS_ZP	BOOL	26.7	FALSE	Status of the zero mark * The bit is only set when counting without main direction. Indicates the zero mark. This is also set when the counter is set to 0 or if it starts counting.
JOB_OVAL	DINT	28.0		Output value for read request.
RES10	BOOL	32.0	FALSE	reserved
RES11	BOOL	32.1	FALSE	reserved
RES_STS	BOOL	32.2	FALSE	Reset status bits: Resets the status bits: <i>STS_CMP</i> , <i>STS_OFLW</i> , <i>STS_ZP</i> . The SFB must be twice called to reset the status bit.

*) Reset with RES_STS



Per channel you may call the SFB in each case with the same instance DB, since the data necessary for the internal operations are stored here. Writing accesses to outputs of the instance DB is not permissible.

Counter request interface

To read/write counter registers the request interface of the SFB 47 may be used. So that a new job may be executed, the previous job must have been finished with *JOB_DONE* = TRUE.

Proceeding

The deployment of the request interface takes place at the following sequence:

1. Edit the following input parameters:

Name	Data type	Address (DB)	Default	Comment
JOB_REQ	BOOL	4.3	FALSE	Initiates the job (edges 0-1) *
JOB_ID	WORD	6.0	0	Job ID: 00h Job without function 01h Writes the count value 02h Writes the load value 04h Writes the comparison value 08h Writes the hysteresis 10h Writes the pulse duration 20h Writes the end value 82h Reads the load value 84h Reads the comparison value 88h Reads the hysteresis 90h Reads the pulse duration A0h Reads the end value
JOB_VAL	DINT	8.0	0	Value for write jobs

*) State remains set also after a CPU STOP-RUN transition.

2. Call the SFB. The job is processed immediately. *JOB_DONE* only applies to SFB run with the result FALSE. *JOB_ERR* = TRUE if an error occurred. Details on the error cause are indicated at *JOB_STAT*.

Name	Data type	Address (DB)	Default	Comment
JOB_DONE	BOOL	22.0	TRUE	New job can be started
JOB_ERR	BOOL	22.1	FALSE	Job error
JOB_STAT	WORD	24.0	0000h	Job error ID 0000h No error 0121h Compare value too low 0122h Compare value too high 0131h Hysteresis too low 0132h Hysteresis too high 0141h Pulse duration too low 0142h Pulse duration too high 0151h Load value too low 0152h Load value too high 0161h Count value too low 0162h Count value too high 01FFh Invalid job ID

3. A new job may be started with *JOB_DONE* = TRUE.
4. A value to be read of a read job may be found in *JOB_OVAL* in the instance DB at address 28.

Permitted value range for *JOB_VAL*

Continuous count:

Job	Valid range
Writing counter directly	-2147483647 ($-2^{31}+1$) ... +2147483646 ($2^{31}-2$)
Writing the load value	-2147483647 ($-2^{31}+1$) ... +2147483646 ($2^{31}-2$)
Writing comparison value	-2147483648 (-2^{31}) ... +2147483647 ($2^{31}-1$)
Writing hysteresis	0 ... 255
Writing pulse duration*	0 ... 510ms

Single/periodic count, no main count direction:

Job	Valid range
Writing counter directly	-2147483647 ($-2^{31}+1$) ... +2147483646 ($2^{31}-2$)
Writing the load value	-2147483647 ($-2^{31}+1$) ... +2147483646 ($2^{31}-2$)
Writing comparison value	-2147483648 (-2^{31}) ... +2147483647 ($2^{31}-1$)
Writing hysteresis	0 ... 255
Writing pulse duration*	0 ... 510ms

Single/periodic count, main count direction up:

Job	Valid range
End value	2 ... +2147483646 ($2^{31}-1$)
Writing counter directly	-2147483648 (-2^{31}) ... end value -2
Writing the load value	-2147483648 (-2^{31}) ... end value -2
Writing comparison value	-2147483648 (-2^{31}) ... end value -1
Writing hysteresis	0 ... 255
Writing pulse duration*	0 ... 510ms

Single/periodic count, main count direction down:

Job	Valid range
Writing counter directly	2 ... +2147483647 ($2^{31}-1$)
Writing the load value	2 ... +2147483647 ($2^{31}-1$)
Writing comparison value	1 ... +2147483647 ($2^{31}-1$)

Job	Valid range
Writing hysteresis	0 ... 255
Writing pulse duration*	0 ... 510ms
*) Only even values allowed. Odd values are automatically rounded.	

Latch function

As soon as during a count process an edge 0-1 is recognized at the "Latch" input of a counter, the recent counter value is stored in the according latch register.

You may access the latch register via *LATCHVAL* of the SFB 47.

A just in *LATCHVAL* loaded value remains after a STOP-RUN transition.

6.5.3 Counter - Functions

Overview

You may count forward and backwards and choose between the following counter functions:

- Count endless, e.g. distance measuring with incremental encoder
- Count once, e.g. count to a maximum limit
- Count periodic, e.g. count with repeated counter process

In the operating modes "Count once" and "Count periodic" you may define a counter range as start and end value via the parameterization. For every counter additional parameterizable functions are available like gate function, comparison, hysteresis and process interrupt.

Main counting direction

Via the parameterization you have the opportunity to define a main counting direction for every counter. If "none" is chosen, the complete counting range is available:

Limits	Valid value range
Lower count limit	-2 147 483 648 (-2^{31})
Upper count limit	+2 147 483 647 ($2^{31}-1$)

Main counting direction forward

Upper restriction of the count range. The counter counts 0 res. *load value* in positive direction until the parameterized *end value* -1 and jumps then back to the load value with the next following encoder pulse.

Main counting direction backwards

Lower restriction of the count range. The counter counts from the parameterized start- res. *load value* in negative direction to the parameterized *end value* +1 and jumps then back to the start value with the next following encoder pulse.

Gate function abort/interrupt

If the HW gate is enabled, only the HW gate may be influenced by the gate functions. An opening and closing of the SW gate only interrupts the count process.

Abort count process

The count process starts after closing and restart of the gate beginning with the *load value*.

Interrupt count process

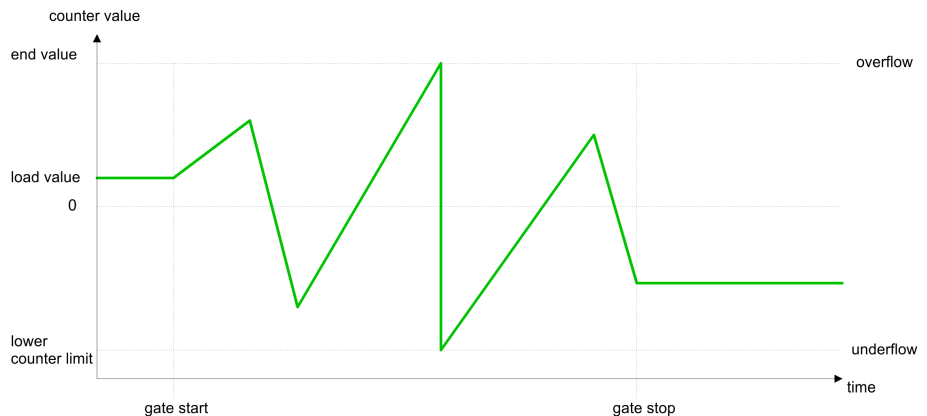
The count process continuous after closing and restart of the gate beginning with the last recent counter value.

Count continuously

In this operating mode, the counter counts from the load value. When the counter counts forward and reaches the upper count limit and another counting pulse in positive direction arrives, it jumps to the lower count limit and counts from there on. When the counter counts backwards and reaches the lower count limit and another counting pulse in negative direction arrives, it jumps to the upper count limit and counts from there on. The count limits are set to the maximum count range.

Limits	Valid value range
Lower count limit	-2 147 483 648 (-2^{31})
Upper count limit	+2 147 483 647 ($2^{31}-1$)

With overflow or underflow the status bits STS_OFLW respectively STS_UFLW are set. These bits remain set until these are reset with RES_STS. If enabled additionally a process interrupt is triggered.



Count Once

No main counting direction

- The counter counts once starting with the *load value*.
- You may count forward or backwards.
- The count limits are set to the maximum count range.
- At over- or underflow at the count limits, the counter jumps to the according other count limit and the internal gate is automatically closed and the status bits STS_OFLW respectively STS_UFLW are set. If enabled additionally a process interrupt is triggered.
- To restart the count process, you have to re-open the internal gate.
- At interrupting gate control, the count process continuous with the last recent *counter value*.
- At aborting gate control, the counter starts with the *load value*.

Limits	Valid value range
Lower count limit	-2 147 483 648 (-2^{31})
Upper count limit	+2 147 483 647 ($2^{31} - 1$)

Interrupting gate control:



Aborting gate control:



Main counting direction forward

- The counter counts starting with the *load value*.
- When the counter reaches the end value -1 in positive direction, it jumps to the load value at the next positive count pulse and the gate is automatically closed.
- To restart the count process, you must create a positive edge of the gate. The counter starts with the load value.

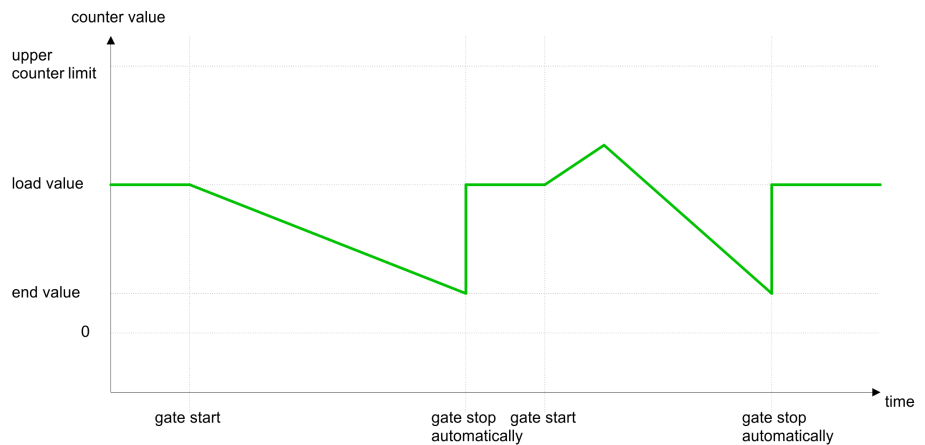
Limits	Valid value range
Limit value	-2 147 483 647 ($-2^{31} + 1$) to +2 147 483 647 ($2^{31} - 1$)
Lower count limit	-2 147 483 648 (-2^{31})



Main counting direction backwards

- The counter counts backwards starting with the *load value*.
- When the counter reaches the end value +1 in negative direction, it jumps to the load value at the next negative count pulse and the gate is automatically closed.
- To restart the count process, you must create a positive edge of the gate. The counter starts with the load value.

Limits	Valid value range
Limit value	-2 147 483 648 (-2^{31}) to +2 147 483 646 ($2^{31} - 2$)
Upper count limit	+2 147 483 647 ($2^{31} - 1$)



Count Periodically

No main counting direction

- The counter counts forward or backwards starting with the *load value*.
- At over- or underrun at the count limits, the counter jumps to the according other count limit and counts from there on.
- The count limits are set to the maximum count range.

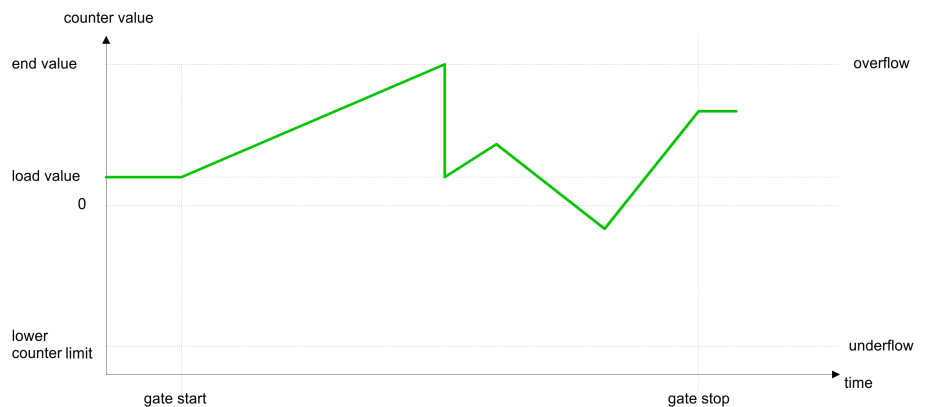
Limits	Valid value range
Lower count limit	-2 147 483 648 (-2^{31})
Upper count limit	+2 147 483 647 ($2^{31} - 1$)



Main counting direction forward

- The counter counts forward starting with the *load value*.
- When the counter reaches the end value -1 in positive direction, it jumps to the *load value* at the next positive count pulse.

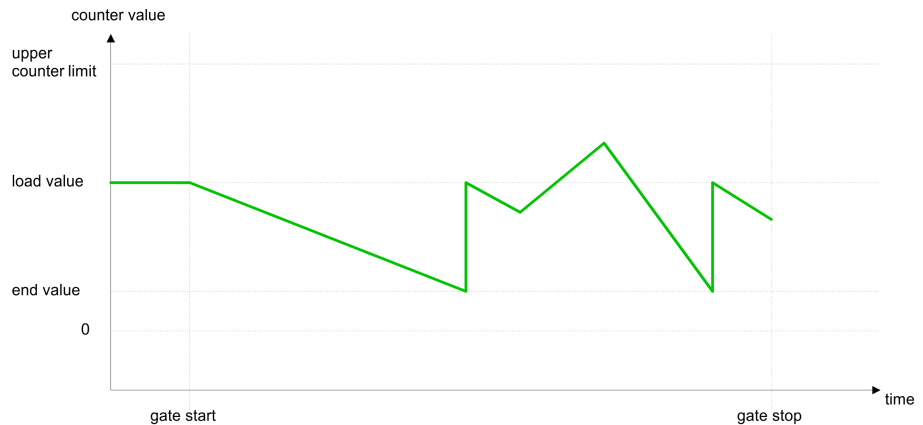
Limits	Valid value range
Limit value	-2 147 483 647 ($-2^{31} + 1$) to +2 147 483 647 ($2^{31} - 1$)
Lower count limit	-2 147 483 648 (-2^{31})



Main counting direction backwards

- The counter counts backwards starting with the *load value*.
- When the counter reaches the *end value* $+1$ in negative direction, it jumps to the *load value* at the next negative count pulse.
- You may exceed the upper count limit.

Limits	Valid value range
Limit value	-2 147 483 648 (-2^{31}) to +2 147 483 646 ($2^{31} - 2$)
Upper count limit	+2 147 483 647 ($2^{31} - 1$)



6.5.4 Counter - Additional functions

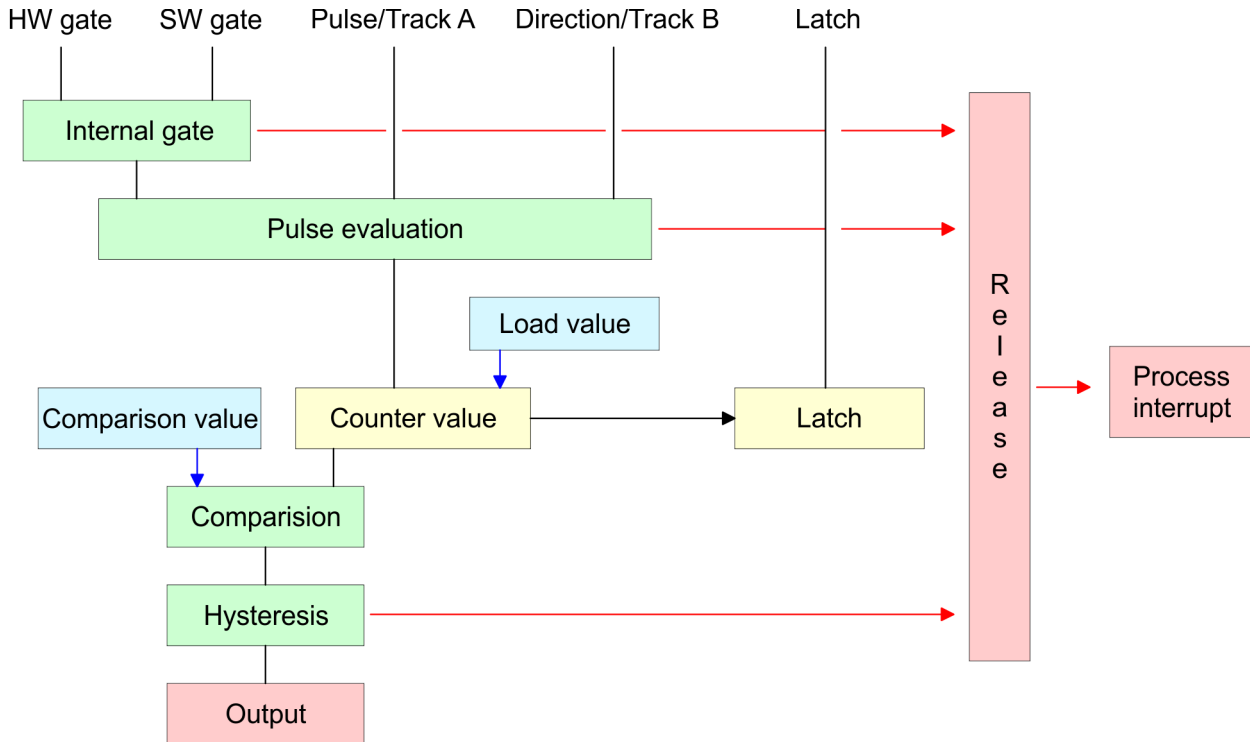
Overview

The following additional functions may be set via the parameterization for every counter:

- Gate function
The gate function serves the start, stop and interrupt of a count function.
- Latch function
An edge 0-1 at the digital input "Latch" stores the recent counter value in the latch register.
- Comparison
You may set a comparison value that activates res. de-activates a digital output res. releases a hardware interrupt depending on the counter value.
- Hysteresis
The setting of a hysteresis avoids for example a high output toggling when the value of an encoder signal shifts around a comparison value.

Schematic structure

The illustration shows how the additional functions influence the counting behaviour. The following pages describe these functions in detail:



Gate function

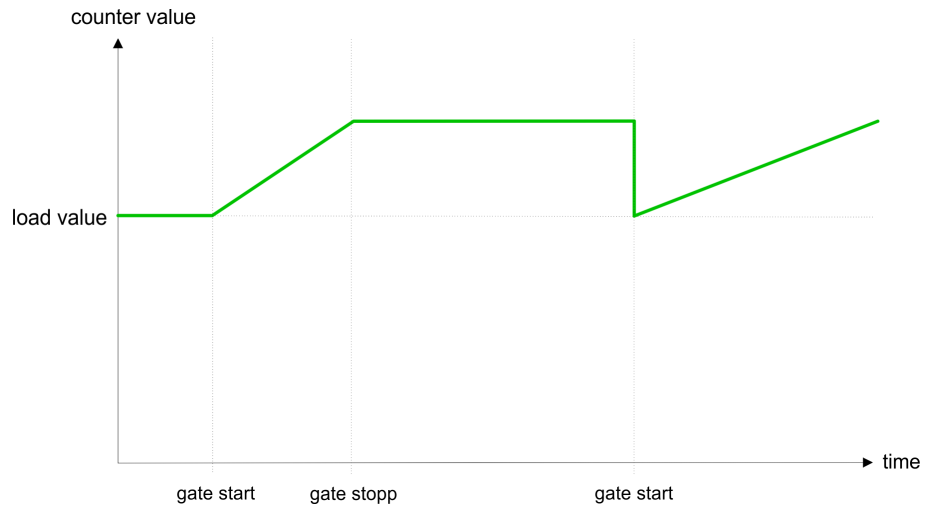
The counter is controlled by the internal gate (i gate). The i gate is the result of logic operation of hardware gate (HW gate) and software gate (SW gate), where the HW gate evaluation may be deactivated by the parameterization.

HW gate: open (activate):	Edge 0-1 at hardware gate _x input of the module
close (deactivate):	Edge 1-0 at hardware gate _x input of the module
SW gate: open (activate):	In application program by setting SW_GATE of the SFB 47
close (deactivate):	In application program by resetting SW_GATE of the SFB 47

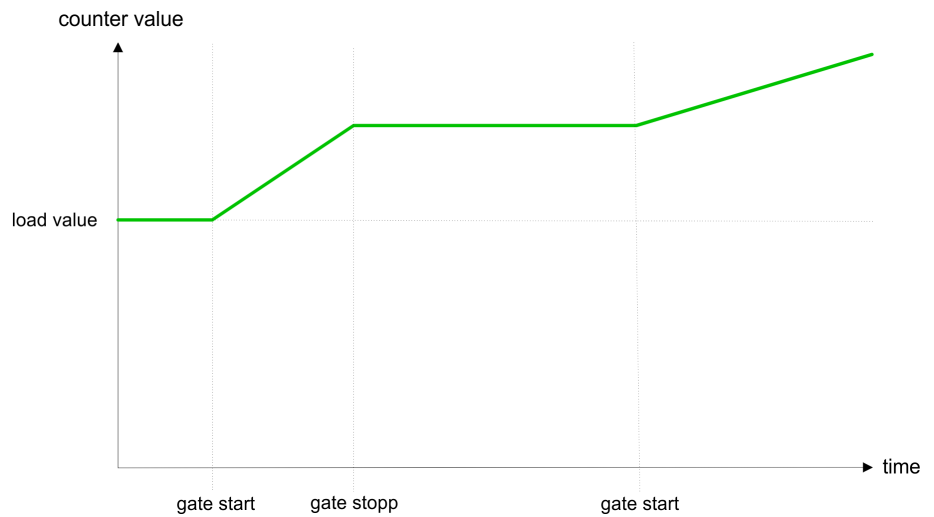
Gate function cancel and stop

The parameterization defines if the gate cancels or stops the counter process.

- At *cancel function* the counter starts counting with the load value after gate restart.



- At *stop function*, the counter continues counting with the last recent counter value after gate restart.



Gate control abort, interruption

How the CPU should react at opening of the SW gate may be set with the parameter *Gate function*. The usage of the HW gate may be determined by the parameter *Hardware gate*.

Gate control via SW gate, canceling (HW gate deactivated, gate function: Cancel count)

SW gate	HW gate	Reaction Counter
edge 0-1	deactivated	Restart with load value

Gate control via SW gate, stopping (HW gate deactivated, gate function: Stop count)

SW gate	HW gate	Reaction Counter
edge 0-1	deactivated	Continue

Gate control via SW/HW gate, canceling (HW gate activated, gate function: Cancel count)

SW gate	HW gate	Reaction Counter
edge 0-1	1	Continue
1	edge 0-1	Restart with load value

Gate control via SW/HW gate, stopping (HW gate activated, gate function: Stop count)

SW gate	HW gate	Reaction Counter
edge 0-1	1	Continue
1	edge 0-1	Continue

Gate control "Count once"

Gate control via SW/HW gate, operating mode "Count once" If the internal gate has been closed automatically it may only be opened again under the following conditions:

SW gate	HW gate	Reaction I gate
1	edge 0-1	1
edge 0-1 (nach edge 0-1 am HW-Tor)	edge 0-1	1

Latch function

As soon as during a count process an edge 0-1 is recognized at the "Latch" input of a counter, the recent counter value is stored in the according latch register. The latch value may be accessed by the parameter LATCHVAL of the SFB 47. A just in LATCHVAL loaded value remains after a STOP-RUN transition.

Comparator

In the CPU a comparison value may be stored that is assigned to the digital output, to the status bit "Status Comparator" STS_CMP and to the hardware interrupt. The digital output may be activated depending on the count value and comparison value. A comparison value may be entered by the parameter assignment screen form respectively by the request interface of the SFB 47.

Characteristics of the output

You pre-define the behavior of the counter output via the parameterization:

- **No comparison**
The output is set as normal output. The SFB input parameter CTRL_DO is effect less. The status bits STS_DO and STS_CMP (Status comparator in the instance DB) remain reset.
- **Count \geq comparison value respectively Count \leq comparison value**
The output remains set as long as the counter value is higher or equal comparison value respectively lower or equal comparison value. For this the control bit must be set. The comparison result is shown by the status bit STS_CMP. This status bit may only be reset if the comparison condition is no longer fulfilled.
- **Pulse at comparison value**
When the counter reaches the comparison value the output is set for the parameterized pulse duration. If you have configured a main count direction the output is only activated when the comparison value is reached with the specified main count direction. For this the control bit CTRL_DO should be set first. The status of the digital output may be shown by the status bit ST_DO. The comparison result is shown by the status bit STS_CMP. This status bit may only be reset if the pulse duration has run off. comparison condition is no longer fulfilled. With pulse time = 0 the output is as set as the comparison condition is fulfilled.

Pulse duration

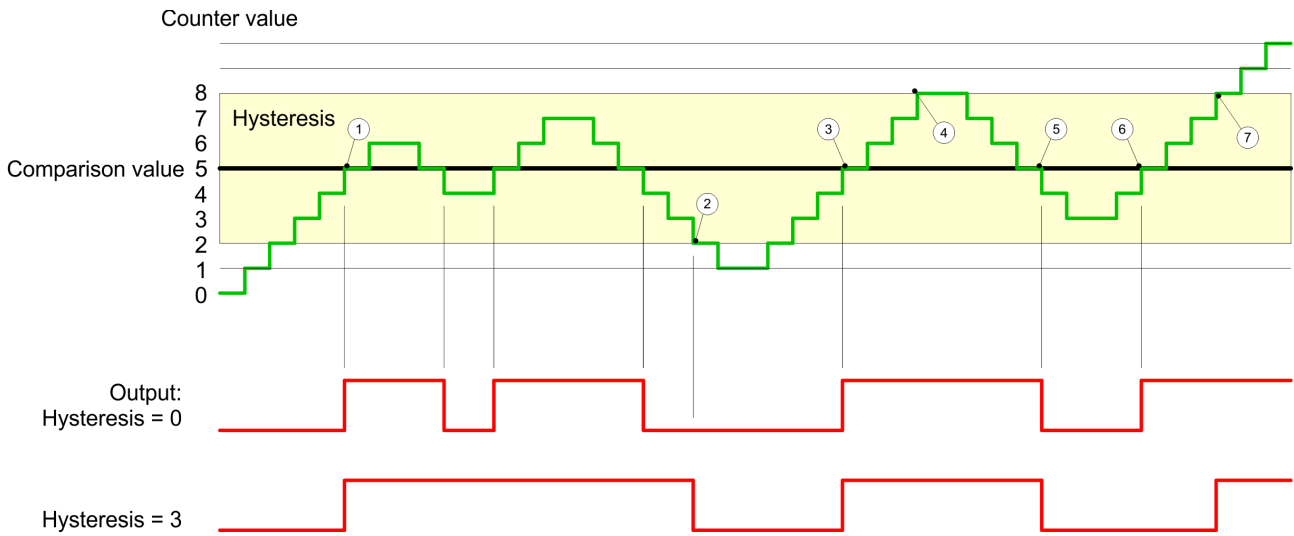
For adaptation to the used actors a pulse duration may be specified. The pulse duration defines how long the output should be set. It may be preset in steps of 2ms between 0 and 510ms. The pulse duration starts with the setting of the according digital output. The inaccuracy of the pulse duration is less than 1ms. There is no past triggering of the pulse duration when the comparison value has been left and reached again during pulse output. A change of the pulse period during runtime is not applied until the next pulse.

Hysteresis

- The *hysteresis* serves the avoidance of many toggle processes of the output and the interrupt, if the *counter value* is in the range of the *comparison value*.
- For the hysteresis you may set a range of 0 to 255.
- The settings 0 and 1 deactivate the *hysteresis*.
- The *hysteresis* influences zero run, comparison, over- and under-flow.
- An activated *hysteresis* remains active after a change. The new *hysteresis* range is activated with the next *hysteresis* event.

The following pictures illustrate the output behavior for *hysteresis* 0 and *hysteresis* 3 for the according conditions:

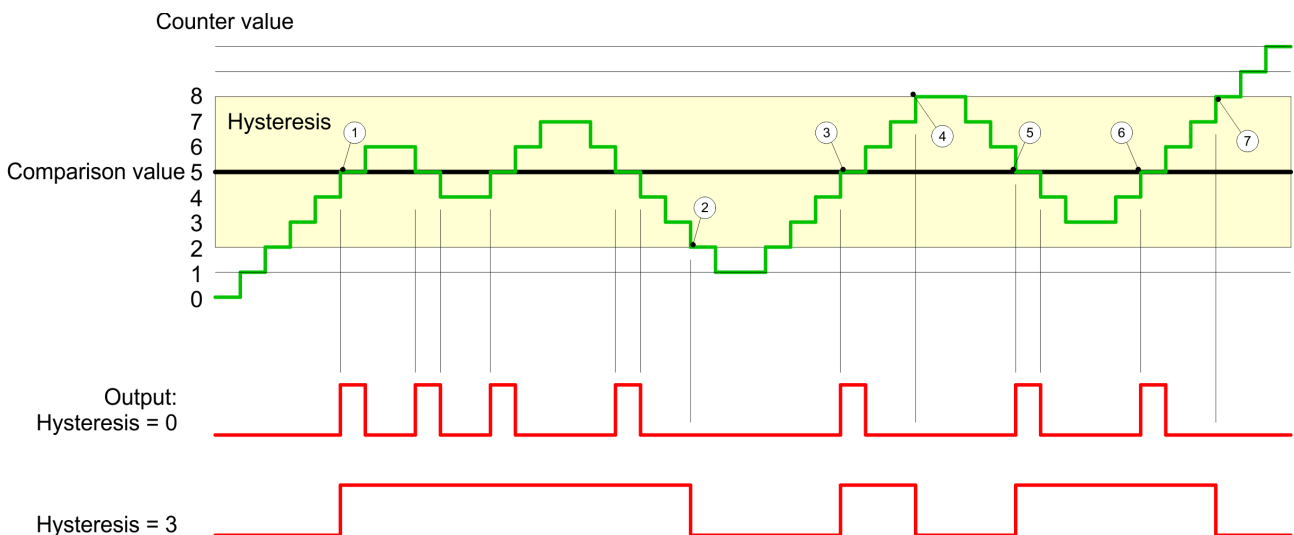
Effect at counter value \geq comparison value



- 1 Counter value \geq comparison value \rightarrow output is set and *hysteresis* activated
- 2 Leave *hysteresis* range \rightarrow output is reset
- 3 Counter value \geq comparison value \rightarrow output is set and *hysteresis* activated
- 4 Leave *hysteresis* range, output remains set for counter value \geq comparison value
- 5 counter value $<$ comparison value and *hysteresis* active \rightarrow output is reset
- 6 counter value \geq comparison value \rightarrow output is not set for *hysteresis* active
- 7 Leave *hysteresis* range, output remains set for counter value \geq comparison value

With reaching the comparison condition the *hysteresis* gets active. At active *hysteresis* the comparison result remains unchanged until the counter value leaves the set *hysteresis* range. After leaving the *hysteresis* range a new *hysteresis* is only activated with again reaching the comparison conditions.

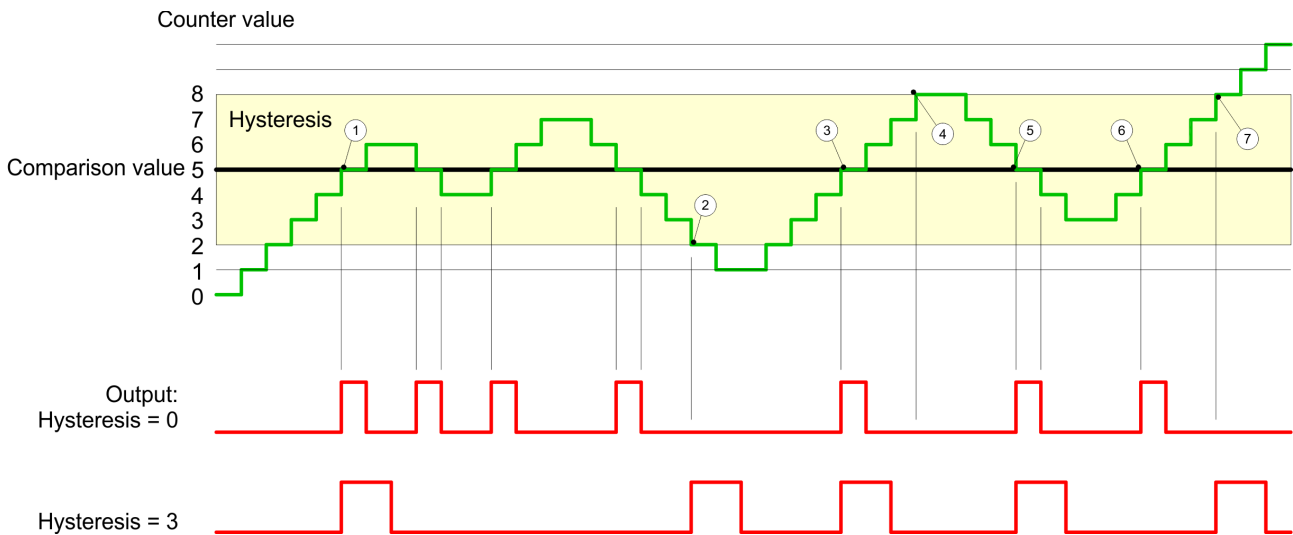
Effect at pulse at comparison value with pulse duration Zero



- 1 Counter value = comparison value → output is set and hysteresis activated
- 2 Leave hysteresis range → output is reset and counter value < comparison value
- 3 Counter value = comparison value → output is set and hysteresis activated
- 4 Output is reset for leaving hysteresis range and counter value > comparison value
- 5 Counter value = comparison value → output is set and hysteresis activated
- 6 Counter value = comparison value and hysteresis active → output remains set
- 7 Leave hysteresis range and counter value > comparison value → output is reset

With reaching the comparison condition the hysteresis gets active. At active hysteresis the comparison result remains unchanged until the counter value leaves the set hysteresis range. After leaving the hysteresis range a new hysteresis is only activated with again reaching the comparison conditions.

Effect at pulse at comparison value with pulse duration not zero



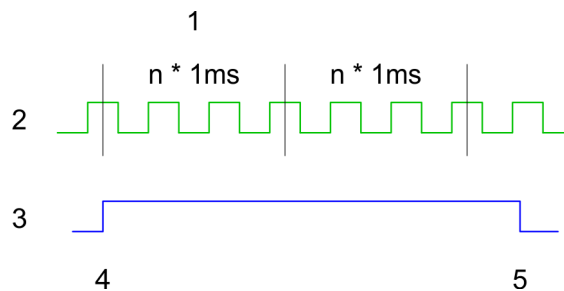
- 1 Counter value = comparison value → pulse of the parameterized pulse duration is put out, the hysteresis is activated and the counting direction stored
- 2 Leaving the hysteresis range contrary to the stored counting direction → pulse of the parameterized pulse duration is put out, the hysteresis is de-activated
- 3 Counter value = comparison value → pulse of the parameterized pulse duration is put out, the hysteresis is activated and the counting direction stored
- 4 Leaving the hysteresis range without changing counting direction → hysteresis is de-activated
- 5 Counter value = comparison value → pulse of the parameterized pulse duration is put out, the hysteresis is activated and the counting direction stored
- 6 Counter value = comparison value and hysteresis active → no pulse
- 7 Leaving the hysteresis range contrary to the stored counting direction → pulse of the parameterized pulse duration is put out, the hysteresis is de-activated

With reaching the comparison condition the *hysteresis* gets active and a pulse of the parameterized duration is put out. As long as the *counter value* is within the *hysteresis* range, no other pulse is put out. With activating the *hysteresis* the counting direction is stored in the module. If the *counter value* leaves the *hysteresis* range *contrary* to the stored counting direction, a pulse of the parameterized duration is put out. Leaving the *hysteresis* range without direction change, no pulse is put out.

6.6 Frequency measurement

6.6.1 Overview

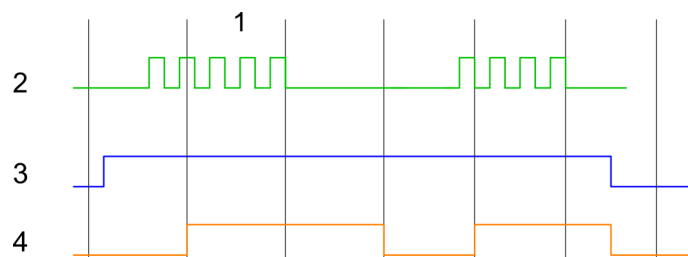
In this operating mode the CPU counts the incoming pulses during a specified integration time and outputs them as frequency value. You can set a value for the integration time between 10ms and 10000ms, in steps of 1 ms. You can set the integration time in the parameter assignment screen forms or you can edit them in the job interface of the SFB FREQUENC (SFB 48).



- 1 Integration time
- 2 Count pulse
- 3 Internal gate (SW gate)
- 4 Start of frequency measurement
- 5 Stop of frequency measurement

Measuring procedure

The measurement is carried out during the integration time and is updated after the integration time has expired. If the period of the measured frequency exceeds the assigned integration time, this means there was no rising edge during the measurement, a value of 0 is returned. The calculated frequency value is supplied in "mHz" units. You can read out this value with the SFB parameter *MEAS_VAL*. The number of activated channels does not influence the max. frequency, which is defined in the technical data.



- 1 Integration time
- 2 Count pulse
- 3 Internal gate (SW gate)
- 4 Calculated frequency

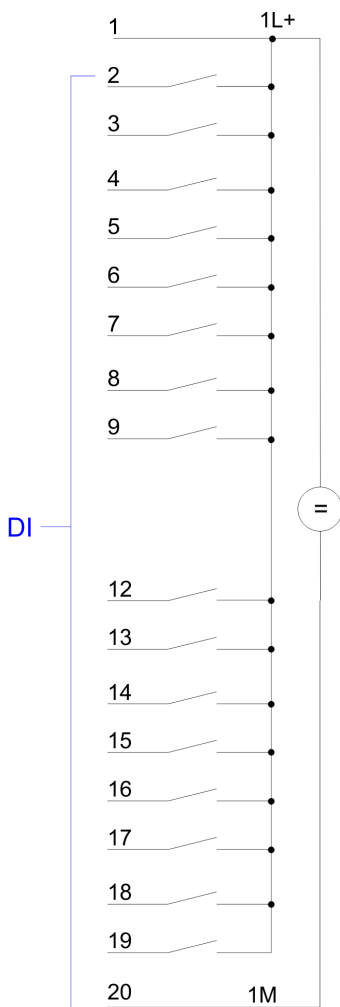


During frequency measurement the count function at the same channel is deactivated.

6.6.2 Inputs for the frequency measurement

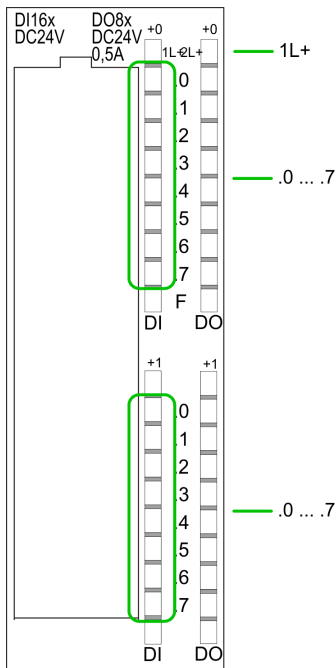
For frequency measurement, connect your signal to be measured at input B.

- Channel 0: Pin 3
- Channel 1: Pin 6



Pin assignment X11: DI

Pin	Assignment
1	1L+ Power supply +DC 24V
2	I+0.0 / Channel 0 (A) / Pulse
3	I+0.1 / Channel 0 (B) / Direction
4	I+0.2 / Channel 0 HW gate
5	I+0.3 / Channel 1 (A) / Pulse
6	I+0.4 / Channel 1 (B) / Direction
7	I+0.5 / Channel 1 HW gate
8	I+0.6
9	I+0.7
10	not used
11	not used
12	I+1.0
13	I+1.1
14	I+1.2
15	I+1.3
16	I+1.4 / Channel 0 Latch
17	I+1.5 / Channel 1 Latch
18	I+1.6
19	I+1.7
20	Ground 1M DI



Status indication X11: DI

- 1L+
 - LED (green)
 - Supply voltage available for DI
- .07
 - LEDs (green)
 - I+0.0 ... I+0.7
 - I+1.0 ... I+1.7
 - Starting with ca. 15V the signal "1" at the input is recognized and the according LED is activated

6.6.3 Parameterization

1. ▶ Start the Siemens SIMATIC Manager with your project and open the hardware configurator.
2. ▶ Place a profile rail.
3. ▶ Configure on slot 2 the Siemens CPU 312C (6ES7 312-5BE03-0AB0 V2.6).
4. ▶ Open the dialog window "Properties" by a double click to the *Count* submodule of the CPU.
5. ▶ As soon as you select the operating mode "Frequency measurement" to the corresponding channel, a dialog window for the frequency measurement is created and displayed and filled with default parameters.
6. ▶ Execute the wished parameterization.
7. ▶ Store the project with 'Station → Save and compile'.
8. ▶ Transfer the project to the CPU.

Parameter overview

In the following the parameters are listed which may be used for frequency measurement configuration during hardware configuration. Parameters, which are not listed here, are ignored by the CPU.

- *General*
Here the short description of the counter function may be found. At Comment information about the module such as purpose may be entered.
- *Addresses*
Here the start address of the counter function is set.

- **Basic parameters**
 Here the interrupts, the counter component should trigger, may be selected. You have the following options:
 - None: There is no interrupt triggered.
 - Process: The counter component triggers a hardware interrupt.
 - Diagnostics and Process: With the CPU the diagnostic interrupt of the digital in-/output periphery is only supported in connection with "hardware interrupt lost".
- **Frequency measurement**
 The following parameters are relevant for frequency measurement. Parameters, which are not listed here, are ignored by the CPU.
 - **Integration time:**
 Integration time for frequency measurement
 Range of values: 10 ... 10000ms
 - **Hardware interrupt:**
 End of measurement (End of integration time)
 When activated, with each end of the integration time, a hardware interrupt is triggered.

6.6.4 SFB 48 - FREQUENC - Frequency measurement

- Description** The SFB 48 is a specially developed block for the CPU 31xSC for frequency measurement.
- The SFB FREQUENC should cyclically be called (e.g. OB 1) for controlling the frequency measurement.
 - The SFB is to be called with the corresponding instance DB. Here the parameters of the SFB are stored.
 - Among others the SFB 48 contains a request interface. Hereby you get read and write access to the registers of the frequency meter.
 - So that a new job may be executed, the previous job must have been finished with *JOB_DONE* = TRUE.
 - Per channel you may call the SFB in each case with the same instance DB, since the data necessary for the internal operational are stored here. Writing accesses to outputs of the instance DB is not permissible.
 - With the SFB FREQUENC (SFB 48) you have following functional options:
 - Start/Stop the frequency meter via software gate SW_GATE
 - Read the status bit
 - Read the evaluated frequency
 - Request to read/write internal registers of the frequency meter.

Parameters

Name	Declaration	Data type	Address (Inst.-DB)	Default value	Comment
LADDR	INPUT	WORD	0.0	300h	This parameter is not evaluated. Always the internal I/O periphery is addressed.
CHANNEL	INPUT	INT	2.0	0	Channel number

Name	Declaration	Data type	Address (Inst.-DB)	Default value	Comment
SW_GATE	INPUT	BOOL	4.0	FALSE	Enables the Software gate
JOB_REQ	INPUT	BOOL	4.3	FALSE	Initiates the job (edge 0-1)
JOB_ID	INPUT	WORD	6.0	0	Job ID
JOB_VAL	INPUT	DINT	8.0	0	Value for write jobs
STS_GATE	OUTPUT	BOOL	12.0	FALSE	Status of the internal gate
MEAS_VAL	OUTPUT	DINT	14.0	0	Evaluated frequency
JOB_DONE	OUTPUT	BOOL	22.0	TRUE	New job can be started.
JOB_ERR	OUTPUT	BOOL	22.1	FALSE	Job error
JOB_STAT	OUTPUT	WORD	24.0	0	Job error ID

Local data only in instance DB

Name	Data type	Address (Instance DB)	Default	Comment
JOB_OVAL	DINT	28.0	-	Output value for read request.



Per channel you may call the SFB in each case with the same instance DB, since the data necessary for the internal operational are stored here. Writing accesses to outputs of the instance DB is not permissible.

Frequency meter request interface

To read/write the registers of the frequency meter the request interface of the SFB 48 may be used.

So that a new job may be executed, the previous job must have been finished with *JOB_DONE* = TRUE.

Proceeding

The deployment of the request interface takes place at the following sequence:

➔ Edit the following input parameters:

Name	Data type	Address (DB)	Default	Comment
JOB_REQ	BOOL	4.3	FALSE	Initiates the job (edges 0-1)
JOB_ID	WORD	6.0	0	Job ID: 00h Job without function 04h Writes the integration time 84h Read the integration time
JOB_VAL	DINT	8.0	0	Value for write jobs. Permitted value for integration time: 10 ... 10000ms

➔ Call the SFB. The job is processed immediately. *JOB_DONE* only applies to SFB run with the result FALSE. *JOB_ERR* = TRUE if an error occurred. Details on the error cause are indicated at *JOB_STAT*.

Name	Data type	Address (DB)	Default	Comment
JOB_DONE	BOOL	22.0	TRUE	New job can be started
JOB_ERR	BOOL	22.1	FALSE	Job error
JOB_STAT	WORD	24.0	0000h	Job error ID 0000h No error 0221h Integration time too low 0222h Integration time too high 02FFh Invalid job ID 8001h Parameter error 8009h Channel no. not valid

1. ➔ A new job may be started with *JOB_DONE* = TRUE.
2. ➔ A value to be read of a read job may be found in *JOB_OVAL* in the instance DB at address 28.

Channel no. not valid

(8009h and Parameter error 8001h)

If you have preset a CHANNEL number greater than 3, the error "Channel no. not valid " (8009h) is reported. if you have preset a CHANNEL number greater than the maximum channel number of the CPU, "Parameter error" (8001h) is reported.

Controlling frequency meter

The frequency meter is controlled by the internal gate (I gate). The I gate is identical to the software gate (SW gate).

SW gate:

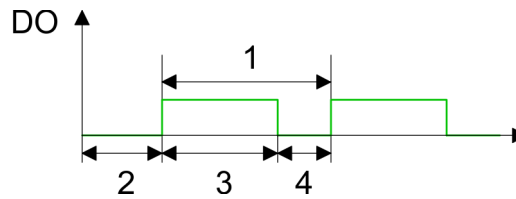
open (activate): In the user program by setting *SW_GATE* of SFB 48
 close (deactivate): In the user program by resetting *SW_GATE* of SFB 48

6.7 Pulse width modulation - PWM

6.7.1 Overview

PWM

With the pulse width modulation (PWM) by presetting of time parameters the CPU evaluates a pulse sequence with according pulse/break ratio and issues it via the according output channel.



- 1 Period duration
- 2 ON delay
- 3 Pulse duration
- 4 Pulse pause



During pulse width modulation the count function at the same channel is deactivated.

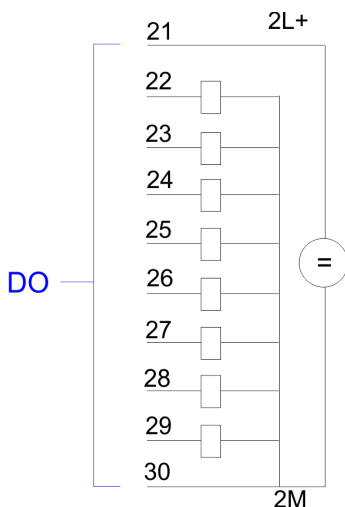
PWM Outputs

For pulse width modulation connect your actuators to the following pins:

Channel 0: Pin 22

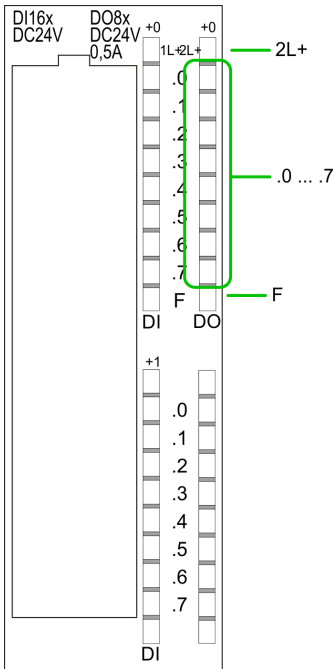
Channel 1: Pin 23

Connect the common ground to pin 30.



Pin assignment X11: DO

Pin	Assignment
21	2L+ Power supply +DC 24V
22	O+0.0 / Channel 0 Output
23	O+0.1 / Channel 1 Output
24	Q+0.2
25	Q+0.3
26	Q+0.4
27	Q+0.5
28	Q+0.6
29	Q+0.7
30	Ground 2M DO
31 ... 40	not used



Status indication X11: DO

- 2L+
 - LED (green)
 - Supply voltage available for DO
- .07
 - LEDs (green)
 - Q+0.0 ... Q+0.7
 - The according LED is on at active output
- F
 - LED (red)
 - Overload or short circuit error

6.7.2 Parameterization

1. ▶ Start the Siemens SIMATIC Manager with your project and open the hardware configurator.
2. ▶ Place a profile rail.
3. ▶ Configure at slot 2 the Siemens CPU 312C (6ES7 312-5BE03-0AB0 V2.6).
4. ▶ Open the dialog window "Properties" by a double click to the *Count* submodule of the CPU.
5. ▶ As soon as you select the operating mode "Pulse width modulation" to the corresponding channel, a dialog window for the pulse width modulation is created and displayed and filled with default parameters.
6. ▶ Execute the wished parameterization.
7. ▶ Save the project with '*Station → Save and compile*'.
8. ▶ Transfer the project to the CPU.

6.7.2.1 Parameter overview

In the following the parameters are listed which may be used for pulse width modulation configuration during hardware configuration. Parameters, which are not listed here, are ignored by the CPU.

General

Here the short description of the counter function may be found. At *Comment* information about the module such as purpose may be entered.

Addresses

Here the start address of the counter function is set.

Basic parameters

Here the interrupts, the counter function should trigger, may be selected. You have the following options:

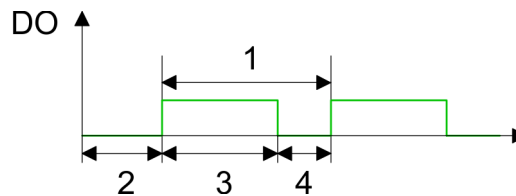
- None: There is no interrupt triggered.
- Process: The counter component triggers a hardware interrupt.
- Diagnostics and Process: With the CPU the diagnostic interrupt of the digital in-/output periphery is only supported in connection with "hardware interrupt lost".



There are no interrupts with the pulse width function.

Pulse width modulation

The following parameters are relevant for pulse width modulation. Parameters, which are not listed here, are ignored by the CPU.



- 1 Period duration
- 2 ON delay
- 3 Pulse duration
- 4 Pulse pause

Output format

- Here select the range of values of the output value. With this the CPU calculates the pulse duration:

Output format	Range of values	Pulse duration
Per mil (Default)	0 ... 1000	$(\text{Output value} / 1000) \times \text{Period duration}$
S7 analog value	0 ... 27648	$(\text{Output value} / 27648) \times \text{Period duration}$

Time base

- Set the time base, which is valid for resolution and the range of values of period duration, minimum pulse duration and on-delay.
- If you have checked the "1ms" button times with a resolution of 1ms can be set.
- If you have checked the "0.1ms" button times with a resolution of 0.1ms can be set.
- Default: "0.1 ms"

On-delay

- Enter a value for the delay time between the start of an output sequence and the output of the pulse. The pulse sequence is output on the output channel after the on-delay has expired.
- Range of values: 0 ... 65535ms respectively 0 ... 6553.5ms

Period (Period duration)

- With the period duration the length of the output sequence (pulse duration/pulse pause) is defined
- Range of values: 1 ... 65535ms respectively 0.4 ... 6553.5ms
- Default value: 20000

Minimum pulse duration

- You can set a minimum pulse time for the attenuation of short output pulses/pauses. This suppresses all pulses or pauses shorter than the minimum pulse time. Thus you may filter very small pulses (spikes), which are not noted from the periphery anymore.
- Range of values:
0 ... Period duration/2 * 1ms respectively
2 ... Period duration/2 * 0.1ms
- Default: 2

6.7.3 SFB 49 - PULSE - Pulse width modulation

Description

The SFB 49 is a specially developed block for the CPU 31xSC for pulse width modulation

- The SFB PULSE should cyclically be called (e.g. OB 1) for controlling the frequency measurement.
- The SFB is to be called with the corresponding instance DB. Here the parameters of the SFB are stored.
- Among others the SFB 49 contains a request interface. Hereby you get read and write access to the registers of the pulse width modulation.
- So that a new job may be executed, the previous job must have been finished with *JOB_DONE* = TRUE.
- Per channel you may call the SFB in each case with the same instance DB, since the data necessary for the internal operational are stored here. Writing accesses to outputs of the instance DB is not permissible.
- With the SFB PULSE (SFB 49) you have following functional options:
 - Start/Stop the pulse width modulation via software gate *SW_GATE*
 - Enabling/controlling of the PWM output
 - Read status bits
 - Request to read/write internal registers of the pulse width modulation

Parameters

Name	Declaration	Data type	Address (Inst.-DB)	Default value	Comment
LADDR	INPUT	WORD	0.0	300h	This parameter is not evaluated. Always the internal I/O periphery is addressed.
CHANNEL	INPUT	INT	2.0	0	Channel number
SW_EN	INPUT	BOOL	4.0	FALSE	Enables the Software gate
OUTP_VAL	INPUT	INT	6.0	0	Output value

Pulse width modulation - PWM > SFB 49 - PULSE - Pulse width modulation

Name	Declaration	Data type	Address (Inst.-DB)	Default value	Comment
JOB_REQ	INPUT	BOOL	8.0	FALSE	Initiates the job (edge 0-1)
JOB_ID	INPUT	WORD	10.0	0	Job ID
JOB_VAL	INPUT	DINT	12.0	0	Value for write jobs
STS_EN	OUTPUT	BOOL	16.0	FALSE	Status of the internal gate
JOB_DONE	OUTPUT	BOOL	16.3	TRUE	New job can be started.
JOB_ERR	OUTPUT	BOOL	16.4	FALSE	Job error
JOB_STAT	OUTPUT	WORD	18.0	0	Job error ID

Local data only in Instance DB

Name	Data type	Address (Instance DB)	Default	Comment
JOB_OVAL	DINT	20.0	-	Output value for read request.



Per channel you may call the SFB in each case with the same instance DB, since the data necessary for the internal operational are stored here. Writing accesses to outputs of the instance DB is not permissible.

PWM Request interface

To read/write the registers of the pulse width modulation the request interface of the SFB 49 may be used.

So that a new job may be executed, the previous job must have been finished with `JOB_DONE = TRUE`.

Proceeding

The deployment of the request interface takes place at the following sequence:

→ Edit the following input parameters:

Name	Data type	Address (DB)	Default	Comment
JOB_REQ	BOOL	8.0	FALSE	Initiates the job (edges 0-1)
JOB_ID	WORD	10.0	0	Job ID: 00h Job without function 01h write period duration 02h write on-delay 04h write minimum pulse duration 81h read period duration 82h read on-delay 84h read minimum pulse duration
JOB_VAL	DINT	8.0	0	Value for write jobs. -2147483648 (-2^{31}) to +2147483647 ($2^{31}-1$)

→ Call the SFB. The job is processed immediately. *JOB_DONE* only applies to SFB run with the result FALSE. *JOB_ERR* = TRUE if an error occurred. Details on the error cause are indicated at *JOB_STAT*.

Name	Data type	Address (DB)	Default	Comment
JOB_DONE	BOOL	22.0	TRUE	New job can be started
JOB_ERR	BOOL	22.1	FALSE	Job error
JOB_STAT	WORD	24.0	0000h	Job error ID 0000h No error 0411h Period duration time too low 0412h Period duration time too high 0421h On-delay too low 0422h On-delay too high 0431h Minimum pulse duration too low 0432h Minimum pulse duration too high 04FFh Invalid job ID 8001h Parameter error 8009h Channel no. not valid

1. ▶ A new job may be started with `JOB_DONE = TRUE`.
2. ▶ A value to be read of a read job may be found in `JOB_OVAL` in the instance DB at address 28.

Channel no. not valid (8009h) and Parameter error (8001h)

If you have preset a CHANNEL number greater than 3, the error "Channel no. not valid" (8009h) is reported. If you have preset a CHANNEL number greater than the maximum channel number of the CPU, "Parameter error" (8001h) is reported.

Controlling PWM

The pulse width modulation is controlled by the internal gate (I gate). The I gate is identical to the software gate (SW gate).

SW gate:

open (activate): In the user program by setting `SW_EN` of SFB 49

close (deactivate): In the user program by resetting `SW_EN` of SFB 49



If values during the PWM output are changed, the new values will be issued until the beginning of a new period. A just started period runs always to the end!

6.8 Diagnostic and interrupt

Overview

The parameterization allows you to define the following trigger for a hardware interrupt that may initialize a diagnostic interrupt:

Counter function

- Edge at a digital input
- Opening the HW gate (at opened SW gate)
- Closing the HW gate (at opened SW gate)
- Reaching the comparison value
- Overflow respectively at overrun upper counter limit
- Underflow respectively at underrun lower counter limit

Frequency measurement

- Edge at a digital input
- End of measurement (end of the integration time)

Pulse width modulation

- There is no process interrupt available

6.8.1 Process interrupt

Activation



The activation of the process interrupt occurs only if at the same time the counter output is activated and "Diagnostics +Process" is set in the basic parameters.

Process interrupt

A process interrupt causes a call of the OB 40. Within the OB 40 you may find the logical basic address of the module that initialized the process interrupt by using the Local word 6. More detailed information about the initializing event is to find in the *local double word 8*.

Local double word 8 of OB 40 at counter function

Local byte	Bit 7...0
8	<ul style="list-style-type: none"> ■ Bit 0: Edge at I+0.0 ■ Bit 1: Edge at I+0.1 ■ Bit 2: Edge at I+0.2 ■ Bit 3: Edge at I+0.3 ■ Bit 4: Edge at I+0.4 ■ Bit 5: Edge at I+0.5 ■ Bit 6: Edge at I+0.6 ■ Bit 7: Edge at I+0.7
9	<ul style="list-style-type: none"> ■ Bit 0: Edge at I+1.0 ■ Bit 1: Edge at I+1.1 ■ Bit 7 ... 2: reserved
10	<ul style="list-style-type: none"> ■ Bit 0: Gate counter 0 open (activated) ■ Bit 1: Gate counter 0 closed ■ Bit 2: Over-/underflow/end value counter 0 ■ Bit 3: Counter 0 reached comparison value ■ Bit 4: Gate counter 1 open (activated) ■ Bit 5: Gate counter 1 closed ■ Bit 6: Over-/underflow/ end value counter 1 ■ Bit 7: Counter 1 reached comparison value
11	<ul style="list-style-type: none"> ■ Bit 7 ... 0: reserved

Local double word 8 of OB 40 at frequency measurement

Local byte	Bit 7...0
8	<ul style="list-style-type: none"> ■ Bit 0: Edge at I+0.0 ■ Bit 1: Edge at I+0.1 ■ Bit 2: Edge at I+0.2 ■ Bit 3: Edge at I+0.3 ■ Bit 4: Edge at I+0.4 ■ Bit 5: Edge at I+0.5 ■ Bit 6: Edge at I+0.6 ■ Bit 7: Edge at I+0.7
9	<ul style="list-style-type: none"> ■ Bit 0: Edge at I+1.0 ■ Bit 1: Edge at I+1.1 ■ Bit 7 ... 2: reserved

Local byte	Bit 7...0
10	<ul style="list-style-type: none"> ■ Bit 0: End of measurement channel 0 (end of the integration time) ■ Bit 3 ... 1: reserved ■ Bit 4: End of measurement channel 1 (end of the integration time) ■ Bit 7 ... 5: reserved
11	<ul style="list-style-type: none"> ■ Bit 7 ... 0: reserved

6.8.2 Diagnostic interrupt

Activation

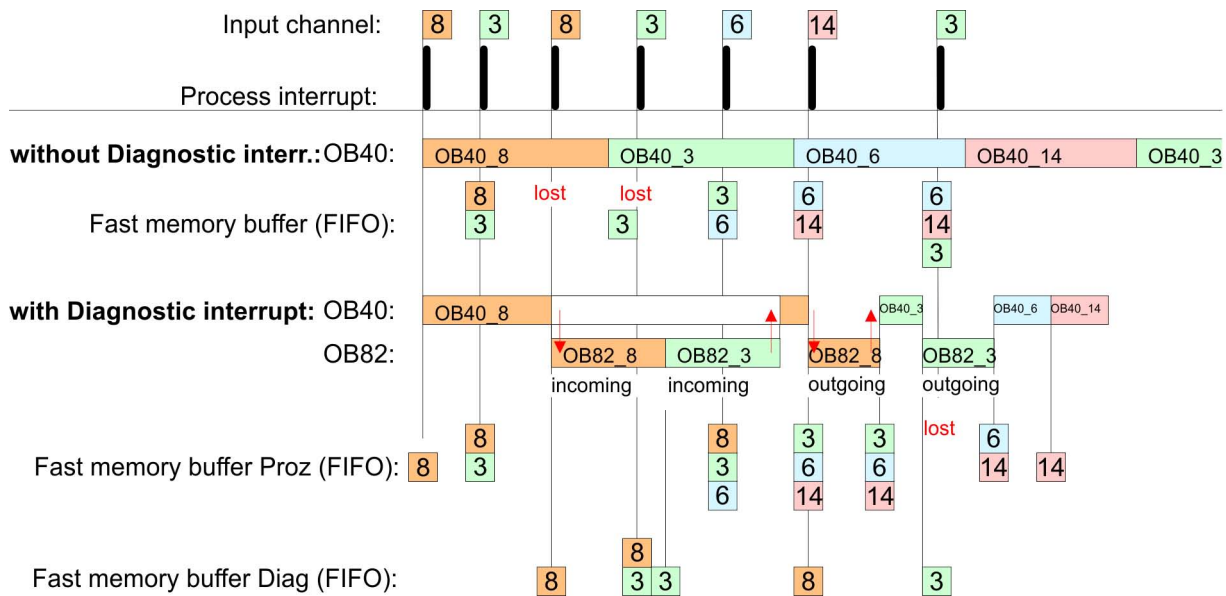


Please consider that diagnostic interrupts are enabled only if you have selected one of the technology functions (counting, frequency measurement, PWM) and set "Diagnostics+Process interrupt" in the basic parameters.

Function

Via the parameterization (record set 7Fh) you may activate a global diagnostic interrupt for the module. A diagnostic interrupt occurs when during a process interrupt execution in OB 40 another process interrupt is thrown for the same event. The initialization of a diagnostic interrupt interrupts the recent process interrupt execution in OB 40 and branches in OB 82 to diagnostic interrupt processing_{incoming}. If during the diagnostic interrupt processing other events are occurring at other channels that may also cause a process res. diagnostic interrupt, these are interim stored. After the end of the diagnostic interrupt processing at first all interim stored diagnostic interrupts are processed in the sequence of their occurrence and then all process interrupts. If a channel where currently a diagnostic interrupt_{incoming} is processed res. interim stored initializes further process interrupts, these get lost. When a process interrupt for which a diagnostic interrupt_{incoming} has been released is ready, the diagnostic interrupt processing is called again as diagnostic interrupt_{outgoing}. All events of a channel between diagnostic interrupt_{incoming} and diagnostic interrupt_{outgoing} are not stored and get lost. Within this time window (1. diagnostic interrupt_{incoming} until last diagnostic interrupt_{outgoing}) the SF-LED of the CPU is on. Additionally for every diagnostic interrupt_{incoming/outgoing} an entry in the diagnostic buffer of the CPU occurs.

Example:



Diagnostic interrupt processing

Every OB 82 call causes an entry in the diagnostic buffer of the CPU containing error cause and module address. By using the SFC 59 you may read the diagnostic bytes. At de-activated diagnostic interrupt you have access to the last recent diagnostic event. If you've activated the diagnostic function in your hardware configuration, the contents of record set 0 are already in the local double word 8 when calling the OB 82. The SFC 59 allows you to also read the record set 1 that contains additional information. After leaving the OB 82 a clear assignment of the data to the last diagnostic interrupt is not longer possible. The record sets of the diagnostic range have the following structure:

Record set 0 Diagnostic_{incoming}

Byte	Bit 7...0
0	<ul style="list-style-type: none"> ■ Bit 0: set at module failure ■ Bit 1: 0 (fix) ■ Bit 2: set at external error ■ Bit 3: set at channel error ■ Bit 7 ... 4: 0 (fix)
1	<ul style="list-style-type: none"> ■ Bit 3 ... 0: Module class <ul style="list-style-type: none"> - 0101b: Analog - 1111b: Digital ■ Bit 4: Channel information present ■ Bit 7 ... 5: 0 (fix)
2	<ul style="list-style-type: none"> ■ Bit 7 ... 0: 0 (fix)
3	<ul style="list-style-type: none"> ■ Bit 5 ... 0: 0 (fix) ■ Bit 6: Process interrupt lost ■ Bit 7: 0 (fix)

Record set 0 Diagnostic_{outgoing}

After the removing error a diagnostic message_{outgoing} takes place if the diagnostic interrupt release is still active.

Byte	Bit 7...0
0	<ul style="list-style-type: none"> ■ Bit 0: set at module failure ■ Bit 1: 0 (fix) ■ Bit 2: set at external error ■ Bit 3: set at channel error ■ Bit 7 ... 4: 0 (fix)
1	<ul style="list-style-type: none"> ■ Bit 3 ... 0: Module class <ul style="list-style-type: none"> – 0101b: Analog – 1111b: Digital ■ Bit 4: Channel information present ■ Bit 7 ... 5: 0 (fix)
2	00h (fix)
3	00h (fix)



The record set 0 of the counter function, frequency measurement and pulse width modulation has the same structure. There are differences in the structure of record set 1.

Diagnostic record set 1 at counter function

The record set 1 contains the 4byte of the record set 0 and additionally 12byte module specific diagnostic data. The diagnostic bytes have the following assignment:

Byte	Bit 7...0
0 ... 3	Content record set 0 ↪ 'Record set 0 Diagnostic _{incoming} ' on page 135
4	<ul style="list-style-type: none"> ■ Bit 6 ... 0: Channel type (here 70h) <ul style="list-style-type: none"> – 70h: Digital input – 71h: Analog input – 72h: Digital output – 73h: Analog output – 74h: Analog input/output ■ Bit 7: More channel types present <ul style="list-style-type: none"> – 0: no – 1: yes
5	Number of diagnostic bits per channel (here 08h)
6	Number of channels of a module (here 08h)
7	<ul style="list-style-type: none"> ■ Bit 0: Error in channel group 0 (I+0.0 ... I+0.3) ■ Bit 1: Error in channel group 1 (I+0.4 ... I+0.7) ■ Bit 2: Error in channel group 2 (I+1.0 ... I+1.1) ■ Bit 3: reserved ■ Bit 4: Error in channel group 4 (counter 0) ■ Bit 5: Error in channel group 5 (counter 1) ■ Bit 6: reserved ■ Bit 7: reserved

Byte	Bit 7...0
8	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: ... input I+0.0 ■ Bit 1: 0 (fix) ■ Bit 2: ... input I+0.1 ■ Bit 3: 0 (fix) ■ Bit 4: ... input I+0.2 ■ Bit 5: 0 (fix) ■ Bit 6: ... input I+0.3 ■ Bit 7: 0 (fix)
9	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: ... input I+0.4 ■ Bit 1: 0 (fix) ■ Bit 2: ... input I+0.5 ■ Bit 3: 0 (fix) ■ Bit 4: ... input I+0.6 ■ Bit 5: 0 (fix) ■ Bit 6: ... input I+0.7 ■ Bit 7: 0 (fix)
10	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: ... input I+1.0 ■ Bit 1: 0 (fix) ■ Bit 2: ... input I+1.1 ■ Bit 3: 0 (fix) ■ Bit 7 ... 4: reserved
11	<ul style="list-style-type: none"> ■ Bit 7 ... 0: reserved
12	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: ... gate counter 0 closed ■ Bit 1: 0 (fix) ■ Bit 2: ... gate counter 0 opened ■ Bit 3: 0 (fix) ■ Bit 4: ... over-/underflow/end value counter 0 ■ Bit 5: 0 (fix) ■ Bit 6: ... counter 0 reached comparison value ■ Bit 7: 0 (fix)
13	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: ... gate counter 1 closed ■ Bit 1: 0 (fix) ■ Bit 2: ... gate counter 1 opened ■ Bit 3: 0 (fix) ■ Bit 4: ... over-/underflow/end value counter 1 ■ Bit 5: 0 (fix) ■ Bit 6: ... counter 1 reached comparison value ■ Bit 7: 0 (fix)
14	reserved
15	reserved

Diagnostic Record set 1 at frequency measurement

The record set 1 contains the 4byte of the record set 0 and additionally 12byte module specific diagnostic data. The diagnostic bytes have the following assignment:

Byte	Bit 7...0
0 ... 3	Content record set 0 ↪ 'Record set 0 Diagnostic _{incoming} ' on page 135
4	<ul style="list-style-type: none"> ■ Bit 6 ... 0: Channel type (here 70h) <ul style="list-style-type: none"> – 70h: Digital input – 71h: Analog input – 72h: Digital output – 73h: Analog output – 74h: Analog input/output ■ Bit 7: More channel types present <ul style="list-style-type: none"> – 0: no – 1: yes
5	Number of diagnostic bits per channel (here 08h)
6	Number of channels of a module (here 08h)
7	<ul style="list-style-type: none"> ■ Bit 0: Error in channel group 0 (I+0.0 ... I+0.3) ■ Bit 1: Error in channel group 1 (I+0.4 ... I+0.7) ■ Bit 2: Error in channel group 2 (I+1.0 ... I+1.1) ■ Bit 3: reserved ■ Bit 4: Error in channel group 4 (Frequency meter 0) ■ Bit 5: Error in channel group 5 (Frequency meter 1) ■ Bit 6: reserved ■ Bit 7: reserved
8	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: ... input I+0.0 ■ Bit 1: 0 (fix) ■ Bit 2: ... input I+0.1 ■ Bit 3: 0 (fix) ■ Bit 4: ... input I+0.2 ■ Bit 5: 0 (fix) ■ Bit 6: ... input I+0.3 ■ Bit 7: 0 (fix)
9	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: ... input I+0.4 ■ Bit 1: 0 (fix) ■ Bit 2: ... input I+0.5 ■ Bit 3: 0 (fix) ■ Bit 4: ... input I+0.6 ■ Bit 5: 0 (fix) ■ Bit 6: ... input I+0.7 ■ Bit 7: 0 (fix)

Byte	Bit 7...0
10	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: ... input I+1.0 ■ Bit 1: 0 (fix) ■ Bit 2: ... input I+1.1 ■ Bit 3: 0 (fix) ■ Bit 7 ... 4: reserved
11	<ul style="list-style-type: none"> ■ Bit 7 ... 0: reserved
12	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: End of measurement channel 0 (End of integration time) ■ Bit 7 ... 1: 0 (fix)
13	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: End of measurement channel 1 (End of integration time) ■ Bit 7 ... 1: 0 (fix)
14	reserved
15	reserved

Diagnostic record set 1 at pulse width modulation

The record set 1 contains the 4byte of the record set 0 and additionally 12byte module specific diagnostic data. The diagnostic bytes have the following assignment:

Byte	Bit 7...0
0 ... 3	Content record set 0 ↪ 'Record set 0 Diagnostic _{incoming} ' on page 135
4	<ul style="list-style-type: none"> ■ Bit 6 ... 0: Channel type (here 70h) <ul style="list-style-type: none"> – 70h: Digital input – 71h: Analog input – 72h: Digital output – 73h: Analog output – 74h: Analog input/output ■ Bit 7: More channel types present <ul style="list-style-type: none"> – 0: no – 1: yes
5	Number of diagnostic bits per channel (here 08h)
6	Number of channels of a module (here 08h)
7	<ul style="list-style-type: none"> ■ Bit 0: Error in channel group 0 (I+0.0 ... I+0.3) ■ Bit 1: Error in channel group 1 (I+0.4 ... I+0.7) ■ Bit 2: Error in channel group 2 (I+1.0 ... I+1.1) ■ Bit 7 ... 3: reserved

Byte	Bit 7...0
8	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: ... input I+0.0 ■ Bit 1: 0 (fix) ■ Bit 2: ... input I+0.1 ■ Bit 3: 0 (fix) ■ Bit 4: ... input I+0.2 ■ Bit 5: 0 (fix) ■ Bit 6: ... input I+0.3 ■ Bit 7: 0 (fix)
9	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: ... input I+0.4 ■ Bit 1: 0 (fix) ■ Bit 2: ... input I+0.5 ■ Bit 3: 0 (fix) ■ Bit 4: ... input I+0.6 ■ Bit 5: 0 (fix) ■ Bit 6: ... input I+0.7 ■ Bit 7: 0 (fix)
10	Diagnostic interrupt due to "process interrupt lost" at... <ul style="list-style-type: none"> ■ Bit 0: ... input I+1.0 ■ Bit 1: 0 (fix) ■ Bit 2: ... input I+1.1 ■ Bit 3: 0 (fix) ■ Bit 7 ... 4: reserved
11 ... 15	<ul style="list-style-type: none"> ■ Bit 7... 0: reserved

7 Deployment PtP communication

7.1 Fast introduction

General	<p>The CPU has a RS485 interface X3, which is fix set to PtP communication (point to point).</p> <ul style="list-style-type: none"> ■ PtP functionality <ul style="list-style-type: none"> – Using the PtP functionality the RS485 interface is allowed to connect via serial point-to-point connection to different source res. target systems.
Protocols	<p>The protocols res. procedures ASCII, STX/ETX, 3964R, USS and Modbus are supported.</p>
Parametrization	<p>The parametrization of the serial interface happens during runtime using the FC/SFC 216 (SER_CFG). For this you have to store the parameters in a DB for all protocols except ASCII.</p>
Communication	<p>The FCs/SFCs are controlling the communication. Send takes place via FC/SFC 217 (SER_SND) and receive via FC/SFC 218 (SER_RCV). The repeated call of the FC/SFC 217 SER_SND delivers a return value for 3964R, USS and Modbus via RetVal that contains, among other things, recent information about the acknowledgement of the partner station. The protocols USS and Modbus allow to evaluate the receipt telegram by calling the FC/SFC 218 SER_RCV after SER_SND. The FCs/SFCs are included in the consignment of the CPU.</p>
Overview FCs/SFCs for serial communication	<p>The following FCs/SFCs are used for the serial communication:</p>

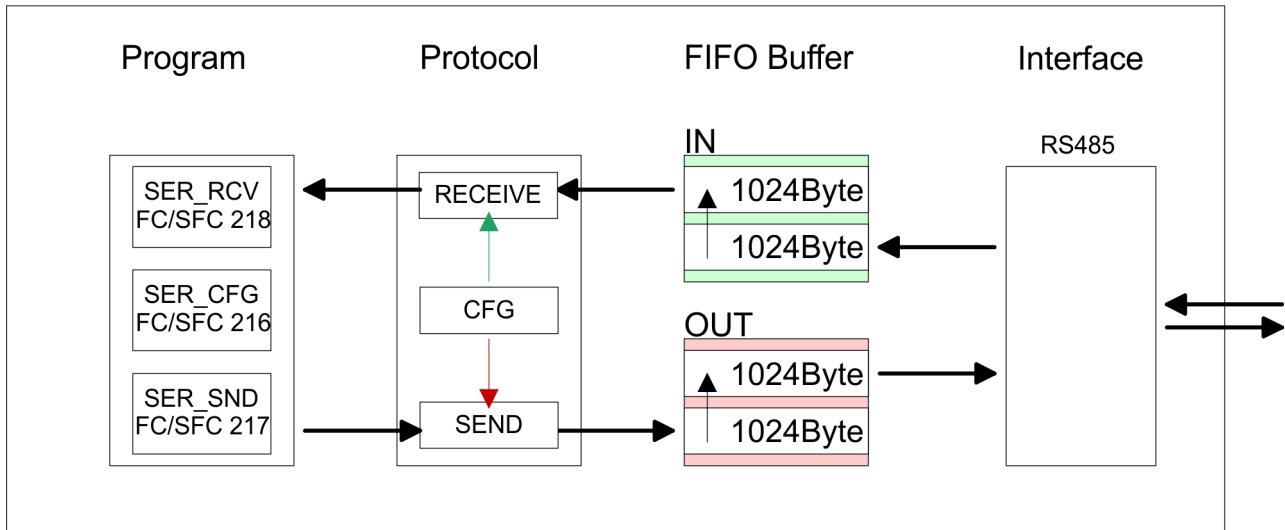
FC/SFC		Description
FC/SFC 216	SER_CFG	RS485 parameterize
FC/SFC 217	SER_SND	RS485 send
FC/SFC 218	SER_RCV	RS485 receive

7.2 Principle of the data transfer

Overview	<p>The data transfer is handled during runtime by using FC/SFCs. The principle of data transfer is the same for all protocols and is shortly illustrated in the following.</p> <ul style="list-style-type: none"> ■ Data, which are written into the according data channel by the CPU, is stored in a FIFO send buffer (first in first out) with a size of 2x1024byte and then put out via the interface. ■ When the interface receives data, this is stored in a FIFO receive buffer with a size of 2x1024byte and can there be read by the CPU. ■ If the data is transferred via a protocol, the embedding of the data to the according protocol happens automatically. ■ In opposite to ASCII and STX/ETX, the protocols 3964R, USS and Modbus require the acknowledgement of the partner.
-----------------	--

- An additional call of the FC/SFC 217 SER_SND causes a return value in RetVal that includes among others recent information about the acknowledgement of the partner.
- Further on for USS and Modbus after a SER_SND the acknowledgement telegram must be evaluated by a call of the FC/SFC 218 SER_RCV.

RS485 PtP communication



7.3 Deployment of RS485 interface for PtP

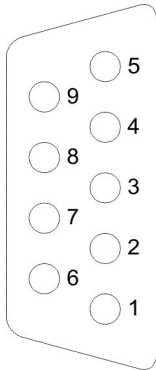
Overview

The RS485 interface from the CPU is fix set to PtP communication. Parameterization and communication happens by means of SFCs.

Properties RS485

- Logical states represented by voltage differences between the two cores of a twisted pair cable
- Serial bus connection in two-wire technology using half duplex mode
- Data communications up to a max. distance of 500m
- Data communication rate up to 115.2kbaud

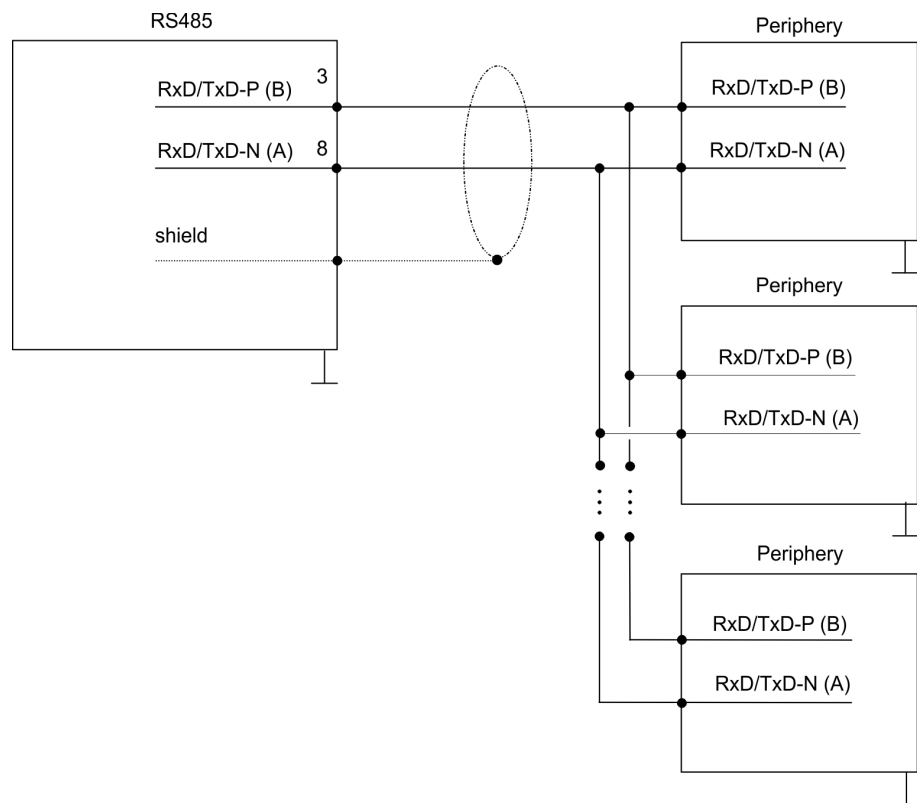
RS485



9pin SubD jack

Pin	RS485
1	n.c.
2	M24V
3	RxD/TxD-P (Line B)
4	RTS
5	M5V
6	P5V
7	P24V
8	RxD/TxD-N (Line A)
9	n.c.

Connection



7.4 Parametrization

7.4.1 FC/SFC 216 - SER_CFG

Description

The parametrization happens during runtime deploying the FC/SFC 216 (SER_CFG). You have to store the parameters for STX/ETX, 3964R, USS and Modbus in a DB.

Parameters

Parameter	Declaration	Data type	Description
PROTOCOL	IN	BYTE	1=ASCII, 2=STX/ETX, 3=3964R
PARAMETER	IN	ANY	Pointer to protocol-parameters
BAUDRATE	IN	BYTE	Number of baudrate
CHARLEN	IN	BYTE	0=5bit, 1=6bit, 2=7bit, 3=8bit
PARITY	IN	BYTE	0=Non, 1=Odd, 2=Even
STOPBITS	IN	BYTE	1=1bit, 2=1.5bit, 3=2bit
FLOWCONTROL	IN	BYTE	1 (fix)
RETVAL	OUT	WORD	Return value (0 = OK)

All time settings for timeouts must be set as hexadecimal value. Find the Hex value by multiply the wanted time in seconds with the baudrate.

Example:

Wanted time 8ms at a baudrate of 19200baud

Calculation: $19200\text{bit/s} \times 0.008\text{s} \approx 154\text{bit} \rightarrow (9\text{Ah})$

Hex value is 9Ah.

PROTOCOL

Here you fix the protocol to be used.

You may choose between:

- 1: ASCII
- 2: STX/ETX
- 3: 3964R
- 4: USS Master
- 5: Modbus RTU Master
- 6: Modbus ASCII Master

PARAMETER (as DB)

At ASCII protocol, this parameter is ignored.

At STX/ETX, 3964R, USS and Modbus you fix here a DB that contains the communication parameters and has the following structure for the according protocols:

Data block at STX/ETX			
DBB0:	STX1	BYTE	(1. Start-ID in hexadecimal)
DBB1:	STX2	BYTE	(2. Start-ID in hexadecimal)
DBB2:	ETX1	BYTE	(1. End-ID in hexadecimal)
DBB3:	ETX2	BYTE	(2. End-ID in hexadecimal)
DBW4:	TIMEOUT	WORD	(max. delay time between 2 telegrams)



The start res. end sign should always be a value <20, otherwise the sign is ignored!

With not used IDs please always enter FFh!

Data block at 3964R

DBB0:	Prio	BYTE	(The priority of both partners must be different)
DBB1:	ConnAttmptNr	BYTE	(Number of connection trials)
DBB2:	SendAttmptNr	BYTE	(Number of telegram retries)
DBB4:	CharTimeout	WORD	(Char. delay time)
DBW6:	ConfTimeout	WORD	(Acknowledgement delay time)

Data block at USS

DBW0:	Timeout	WORD	(Delay time)
-------	---------	------	--------------

Data block at Modbus master

DBW0:	Timeout	WORD	(Respond delay time)
-------	---------	------	----------------------

BAUDRATE

Velocity of data transfer in bit/s (baud)

04h:	1200baud	05h:	1800baud	06h:	2400baud	07h:	4800baud
08h:	7200baud	09h:	9600baud	0Ah:	14400baud	0Bh:	19200baud
0Ch:	38400baud	0Dh:	57600baud	0Eh:	115200baud		

CHARLEN

Number of data bits where a character is mapped to.

0: 5bit	1: 6bit	2: 7bit	3: 8bit
---------	---------	---------	---------

PARITY

The parity is -depending on the value- even or odd. For parity control, the information bits are extended with the parity bit, that amends via its value ("0" or "1") the value of all bits to a defined status. If no parity is set, the parity bit is set to "1", but not evaluated.

0: NONE	1: ODD	2: EVEN
---------	--------	---------

STOPBITS

The stop bits are set at the end of each transferred character and mark the end of a character.

1: 1bit	2: 1.5bit	3: 2bit
---------	-----------	---------

FLOWCONTROL

The parameter *FLOWCONTROL* is ignored. When sending *RTS*=1, when receiving *RTS*=0.

**RETVAL FC/SFC 216
(Return values)**

Return values send by the block:

Error code	Description
0000h	no error
809Ah	Interface not found e. g. interface is used by PROFIBUS In the VIPA SLIO CPU with FeatureSet PTP_NO only the ASCII protocol is configurable. If another protocol is selected the FC/SFC216 also left with this error code.
8x24h	Error at FC/SFC-Parameter x, with x: 1: Error at <i>PROTOCOL</i> 2: Error at <i>PARAMETER</i> 3: Error at <i>BAUDRATE</i> 4: Error at <i>CHARLENGTH</i> 5: Error at <i>PARITY</i> 6: Error at <i>STOPBITS</i> 7: Error at <i>FLOWCONTROL</i>
809xh	Error in FC/SFC parameter value x, where x: 1: Error at <i>PROTOCOL</i> 3: Error at <i>BAUDRATE</i> 4: Error at <i>CHARLENGTH</i> 5: Error at <i>PARITY</i> 6: Error at <i>STOPBITS</i> 7: Error at <i>FLOWCONTROL</i> (parameter is missing)
8092h	Access error in parameter DB (DB too short)
828xh	Error in parameter x of DB parameter, where x: 1: Error 1. parameter 2: Error 2. parameter ...

7.5 Communication**7.5.1 Overview**

The communication happens via the send and receive blocks FC/SFC 217 (*SER_SND*) and FC/SFC 218 (*SER_RCV*). The FCs/SFCs are included in the consignment of the CPU.

7.5.2 FC/SFC 217 - SER_SND

Description This block sends data via the serial interface. The repeated call of the FC/SFC 217 SER_SND delivers a return value for 3964R, USS and Modbus via RETVAL that contains, among other things, recent information about the acknowledgement of the partner station.

The protocols USS and Modbus require to evaluate the receipt telegram by calling the FC/SFC 218 SER_RCV after SER_SND.

Parameters

Parameter	Declaration	Data type	Description
DATAPTR	IN	ANY	Pointer to Data Buffer for sending data
DATALEN	OUT	WORD	Length of data sent
RETVAL	OUT	WORD	Return value (0 = OK)

DATAPTR Here you define a range of the type Pointer for the send buffer where the data to be sent are stored. You have to set type, start and length.

Example:

Data is stored in DB5 starting at 0.0 with a length of 124byte.

DataPtr:=P#DB5.DBX0.0 BYTE 124

DATALEN Word where the number of the sent Bytes is stored.

At **ASCII** if data were sent by means of FC/SFC 217 faster to the serial interface than the interface sends, the length of data to send could differ from the DATALEN due to a buffer overflow. This should be considered by the user program.

With **STX/ETX, 3964R, Modbus** and **USS** always the length set in **DATAPTR** is stored or 0.

RETVAL FC/SFC 217 (Return values)

Return values of the block:

Error code	Description
0000h	Send data - ready
1000h	Nothing sent (data length 0)
20xxh	Protocol executed error free with xx bit pattern for diagnosis
7001h	Data is stored in internal buffer - active (busy)
7002h	Transfer - active
80xxh	Protocol executed with errors with xx bit pattern for diagnosis (no acknowledgement by partner)
90xxh	Protocol not executed with xx bit pattern for diagnosis (no acknowledgement by partner)

Error code	Description
8x24h	Error in FC/SFC parameter x, where x: 1: Error in <i>DATAPTR</i> 2: Error in <i>DATALEN</i>
8122h	Error in parameter <i>DATAPTR</i> (e.g. DB too short)
807Fh	Internal error
809Ah	interface not found e.g. interface is used by PROFIBUS
809Bh	interface not configured

Protocol specific RETVAl values

ASCII

Value	Description
9000h	Buffer overflow (no data send)
9002h	Data too short (0byte)

STX/ETX

Value	Description
9000h	Buffer overflow (no data send)
9001h	Data too long (>1024byte)
9002h	Data too short (0byte)
9004h	Character not allowed

3964R

Value	Description
2000h	Send ready without error
80FFh	NAK received - error in communication
80FEh	Data transfer without acknowledgement of partner or error at acknowledgement
9000h	Buffer overflow (no data send)
9001h	Data too long (>1024byte)
9002h	Data too short (0byte)

USS

Error code	Description
2000h	Send ready without error
8080h	Receive buffer overflow (no space for receipt)
8090h	Acknowledgement delay time exceeded

Error code	Description
80F0h	Wrong checksum in respond
80FEh	Wrong start sign in respond
80FFh	Wrong slave address in respond
9000h	Buffer overflow (no data send)
9001h	Data too long (>1024byte)
9002h	Data too short (<2byte)

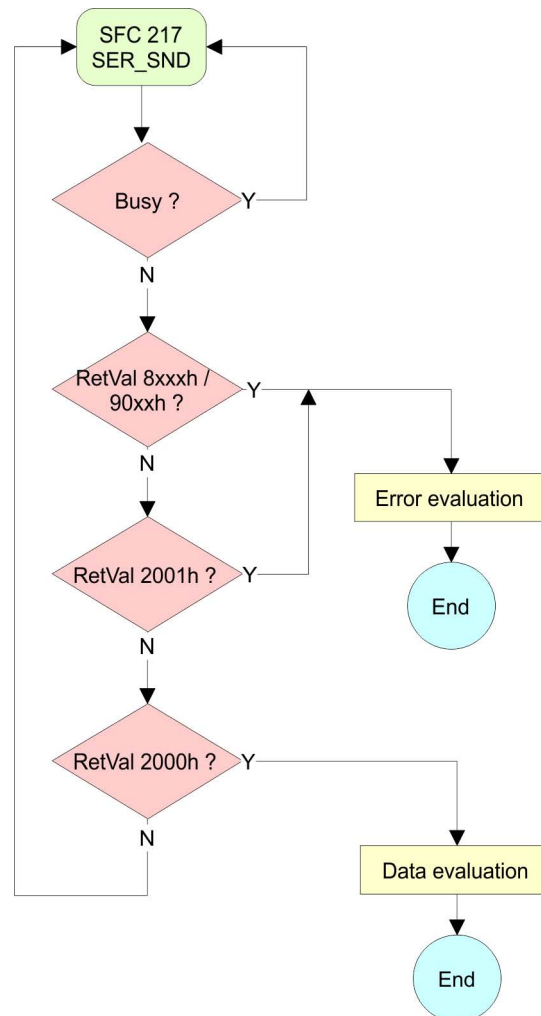
Modbus RTU/ASCII Master

Error code	Description
2000h	Send ready (positive slave respond)
2001h	Send ready (negative slave respond)
8080h	Receive buffer overflow (no space for receipt)
8090h	Acknowledgement delay time exceeded
80F0h	Wrong checksum in respond
80FDh	Length of respond too long
80FEh	Wrong function code in respond
80FFh	Wrong slave address in respond
9000h	Buffer overflow (no data send)
9001h	Data too long (>1024byte)
9002h	Data too short (<2byte)

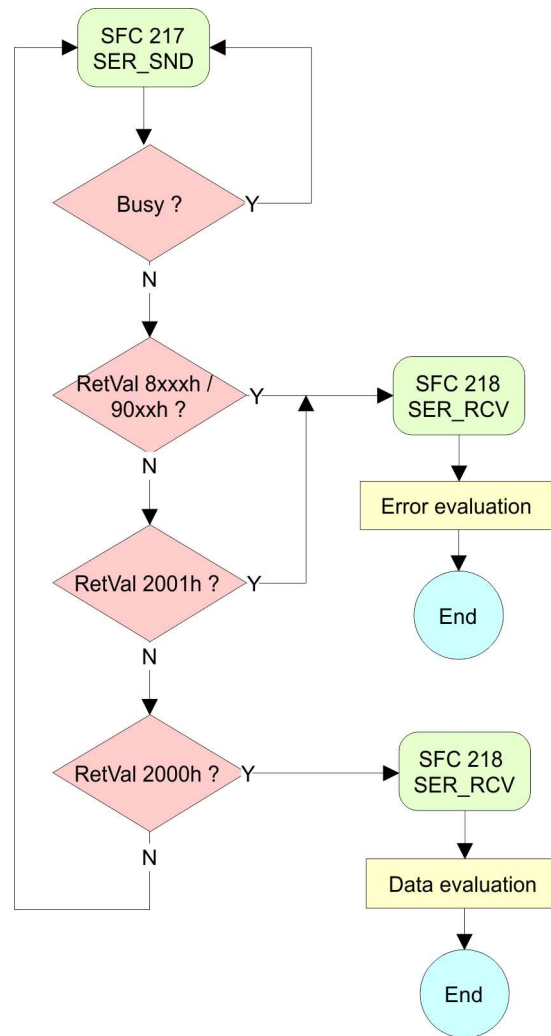
Principles of programming

The following text shortly illustrates the structure of programming a send command for the different protocols.

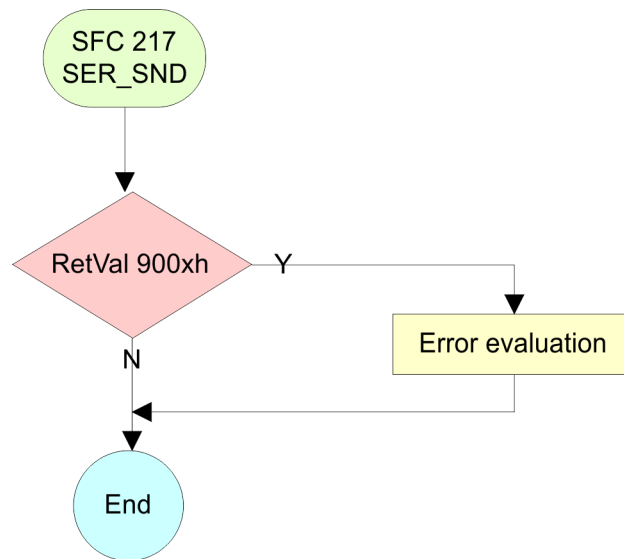
3964R



USS / Modbus



ASCII / STX/ETX



7.5.3 FC/SFC 218 - SER_RCV

Description

This block receives data via the serial interface.

Using the FC/SFC 218 SER_RCV after SER_SND with the protocols USS and Modbus the acknowledgement telegram can be read.

Parameters

Parameter	Declaration	Data type	Description
DATAPTR	IN	ANY	Pointer to Data Buffer for received data
DATALEN	OUT	WORD	Length of received data
ERROR	OUT	WORD	Error Number
RETVAL	OUT	WORD	Return value (0 = OK)

DATAPTR

Here you set a range of the type Pointer for the receive buffer where the reception data is stored. You have to set type, start and length.

Example:

Data is stored in DB5 starting at 0.0 with a length of 124byte.

DataPtr:=P#DB5.DBX0.0 BYTE 124

DATALEN

Word where the number of received Bytes is stored.

At **STX/ETX** and **3964R**, the length of the received user data or 0 is entered.

At **ASCII**, the number of read characters is entered. This value may be different from the read telegram length.

ERROR

This word gets an entry in case of an error.

The following error messages may be created depending on the protocol:

ASCII

Bit	Error	Description
0	overrun	Overflow, a sign couldn't be read fast enough from the interface
1	framing error	Error that shows that a defined bit frame is not coincident, exceeds the allowed length or contains an additional bit sequence (Stop bit error)
2	parity	Parity error
3	overflow	Buffer is full

STX/ETX

Bit	Error	Description
0	overflow	The received telegram exceeds the size of the receive buffer.
1	char	A sign outside the range 20h ... 7Fh has been received.
3	overflow	Buffer is full.

3964R / Modbus RTU/ASCII Master

Bit	Error	Description
0	overflow	The received telegram exceeds the size of the receive buffer.

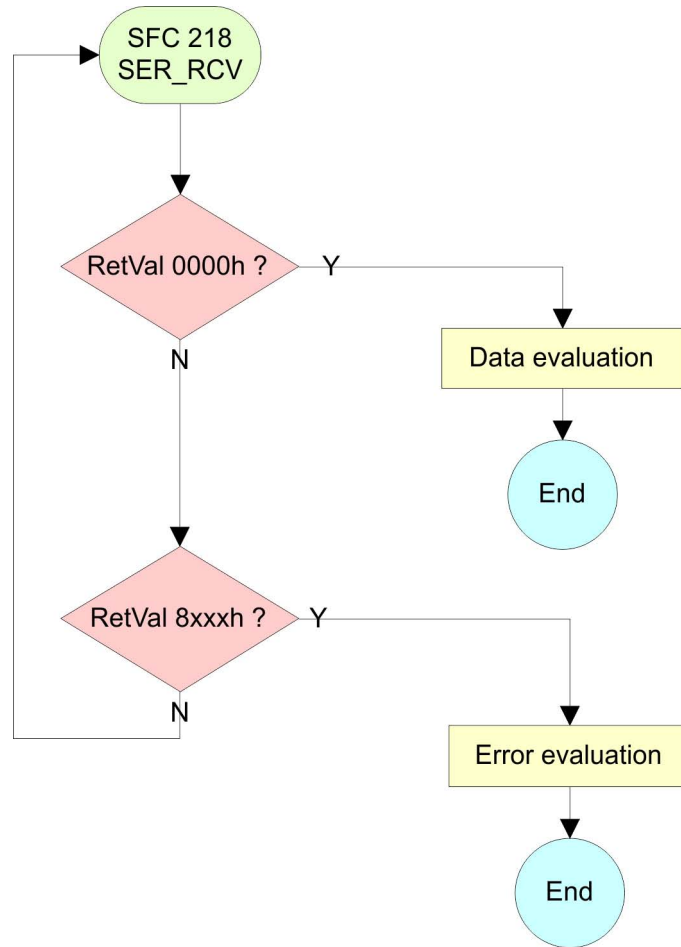
**RETVAL FC/SFC 218
(Return value)**

Return values of the block:

Error code	Description
0000h	no error
1000h	Receive buffer too small (data loss)
8x24h	Error at FC/SFC-Parameter x, with x: 1: Error at <i>DATAPTR</i> 2: Error at <i>DATALEN</i> 3: Error at <i>ERROR</i>
8122h	Error in parameter <i>DATAPTR</i> (e.g. DB too short)
809Ah	Serial interface not found res. interface is used by PROFIBUS
809Bh	Serial interface not configured

Principles of programming

The following picture shows the basic structure for programming a receive command. This structure can be used for all protocols.



7.6 Protocols and procedures

Overview

The CPU supports the following protocols and procedures:

- ASCII communication
- STX/ETX
- 3964R
- USS
- Modbus

ASCII

ASCII data communication is one of the simple forms of data exchange. Incoming characters are transferred 1 to 1. At ASCII, with every cycle the read FC/SFC is used to store the data that is in the buffer at request time in a parameterized receive data block. If a telegram is spread over various cycles, the data is overwritten. There is no reception acknowledgement. The communication procedure has to be controlled by the concerning user application. An according Receive_ASCII FB may be found within the VIPA library in the service area of www.vipa.com.

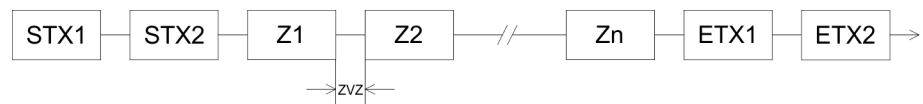
STX/ETX

STX/ETX is a simple protocol with start and end ID, where STX stands for **S**tart of **T**ext and ETX for **E**nd of **T**ext.

- Any data transferred from the periphery must be preceded by a Start followed by the data characters and the end character. Depending of the byte width the following ASCII characters can be transferred: 5bit: not allowed: 6bit: 20...3Fh, 7bit: 20...7Fh, 8bit: 20...FFh.
- The effective data, which includes all the characters between Start and End are transferred to the CPU when the End has been received.
- When data is send from the CPU to a peripheral device, any user data is handed to the FC/SFC 217 (SER_SND) and is transferred with added Start- and End-ID to the communication partner.
- You may work with 1, 2 or no Start- and with 1, 2 or no End-ID.
- If no End-ID is defined, all read characters are transferred to the CPU after a parameterizable character delay time (Timeout).

As Start-res. End-ID all Hex values from 01h to 1Fh are permissible. Characters above 1Fh are ignored. In the user data, characters below 20h are not allowed and may cause errors. The number of Start- and End-IDs may be different (1 Start, 2 End res. 2 Start, 1 End or other combinations). For not used start and end characters you have to enter FFh in the hardware configuration.

Message structure:



3964

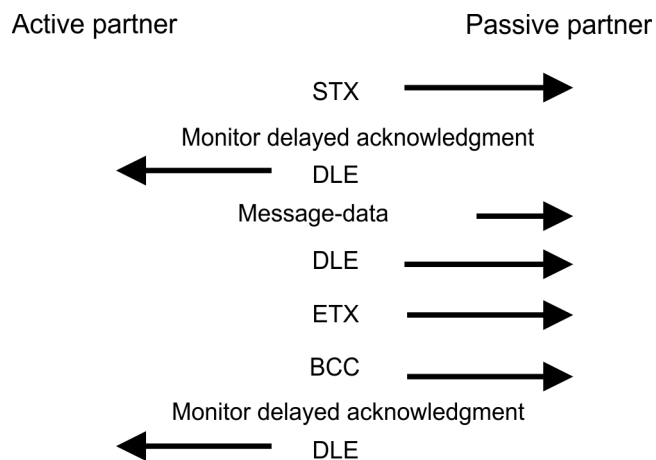
The 3964R procedure controls the data transfer of a point-to-point link between the CPU and a communication partner. The procedure adds control characters to the message data during data transfer. These control characters may be used by the communication partner to verify the complete and error free receipt.

The procedure employs the following control characters:

- STX: **S**tart of **T**ext
- DLE: **D**ata **L**ink **E**scape
- ETX: **E**nd of **T**ext
- BCC: **B**lock **C**heck **C**haracter
- NAK: **N**egative **A**cknowledge

You may transfer a maximum of 255byte per message.

Procedure



When a DLE is transferred as part of the information it is repeated to distinguish between data characters and DLE control characters that are used to establish and to terminate the connection (DLE duplication). The DLE duplication is reversed in the receiving station.

The 3964R procedure requires that a lower priority is assigned to the communication partner. When communication partners issue simultaneous send commands, the station with the lower priority will delay its send command.

USS

The USS protocol (**U**niverselle **s**erielle **S**chnittstelle = universal serial interface) is a serial transfer protocol defined by Siemens for the drive and system components. This allows to build-up a serial bus connection between a superordinated master and several slave systems. The USS protocol enables a time cyclic telegram traffic by presetting a fix telegram length.

The following features characterize the USS protocol:

- Multi point connection
- Master slave access procedure
- Single master system
- Max. 32 participants
- Simple and secure telegram frame

It is essential:

- You may connect 1 master and max. 31 slaves at the bus
- The single slaves are addressed by the master via an address sign in the telegram.
- The communication happens exclusively in half-duplex operation.
- After a send command, the acknowledgement telegram must be read by a call of the FC/SFC 218 SER_RCV.

The telegrams for send and receive have the following structure:

Master slave telegram

STX	LGE	ADR	PKE		IND		PWE		STW		HSW		BCC
02h			H	L	H	L	H	L	H	L	H	L	

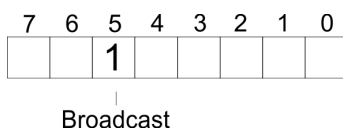
Slave master telegram

STX	LGE	ADR	PKE		IND		PWE		ZSW		HIW		BCC
02h			H	L	H	L	H	L	H	L	H	L	

with

- STX - Start sign
- STW - Control word
- LGE - Telegram length
- ZSW - State word
- ADR - Address
- HSW - Main set value
- PKE - Parameter ID
- HIW - Main effective value
- IND - Index
- BCC - Block Check Character
- PWE - Parameter value

Broadcast with set bit 5 in ADR byte



A request can be directed to a certain slave ore be send to all slaves as broadcast message. For the identification of a broadcast message you have to set bit 5 to 1 in the ADR byte. Here the slave addr. (bit 0 ... 4) is ignored. In opposite to a "normal" send command, the broadcast does not require a telegram evaluation via FC/SFC 218 SER_RCV. Only write commands may be sent as broadcast.

Modbus

- The Modbus protocol is a communication protocol that fixes a hierarchic structure with one master and several slaves.
- Physically, Modbus works with a serial half-duplex connection. There are no bus conflicts occurring, because the master can only communicate with one slave at a time.

- After a request from the master, this waits for a preset delay time for an answer of the slave. During the delay time, communication with other slaves is not possible.
- After a send command, the acknowledgement telegram must be read by a call of the FC/SFC 218 SER_RCV.
- The request telegrams send by the master and the respond telegrams of a slave have the following structure:

Telegram structure

Start sign	Slave address	Function Code	Data	Flow control	End sign
------------	---------------	---------------	------	--------------	----------

Broadcast with slave address = 0

- A request can be directed to a special slave or at all slaves as broadcast message.
- To mark a broadcast message, the slave address 0 is used.
- In opposite to a "normal" send command, the broadcast does not require a telegram evaluation via FC/SFC 218 SER_RCV.
- Only write commands may be sent as broadcast.

ASCII, RTU mode

Modbus offers 2 different transfer modes. The mode selection happens during runtime by using the FC/SFC 216 SER_CFG.

- ASCII mode: Every byte is transferred in the 2 sign ASCII code. The data are marked with a start and an end sign. This causes a transparent but slow transfer.
- RTU mode: Every byte is transferred as one character. This enables a higher data pass through as the ASCII mode. Instead of start and end sign, a time control is used.

Supported Modbus protocols

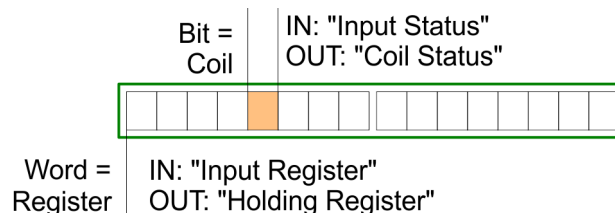
The following Modbus Protocols are supported by the RS485 interface:

- Modbus RTU Master
- Modbus ASCII Master

7.7 Modbus - Function codes

Naming convention

Modbus has some naming conventions:



- Modbus differentiates between bit and word access; bits = "Coils" and words = "Register".
- Bit inputs are referred to as "Input-Status" and bit outputs as "Coil-Status".
- word inputs are referred to as "Input-Register" and word outputs as "Holding-Register".

Range definitions

Normally the access at Modbus happens by means of the ranges 0x, 1x, 3x and 4x.

0x and 1x gives you access to digital bit areas and 3x and 4x to analog word areas.

For the CPs from VIPA is not differentiating digital and analog data, the following assignment is valid:

0x - Bit area for master output data

Access via function code 01h, 05h, 0Fh

1x - Bit area for master input data

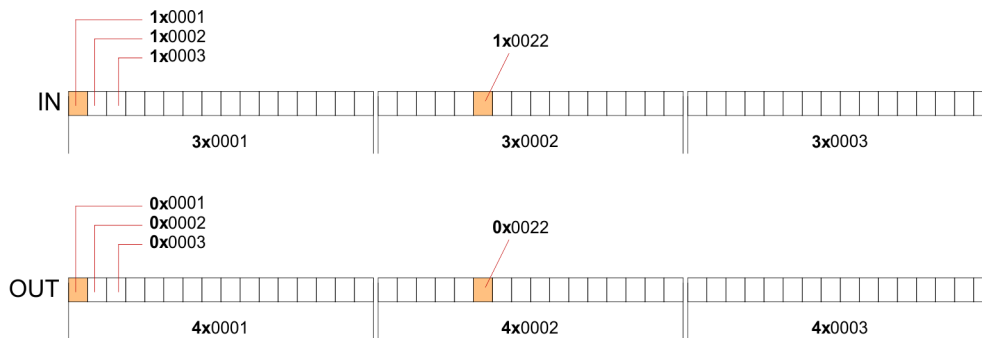
Access via function code 02h

3x - word area for master input data

Access via function code 04h

4x - word area for master output data

Access via function code 03h, 06h, 10h



A description of the function codes follows below.

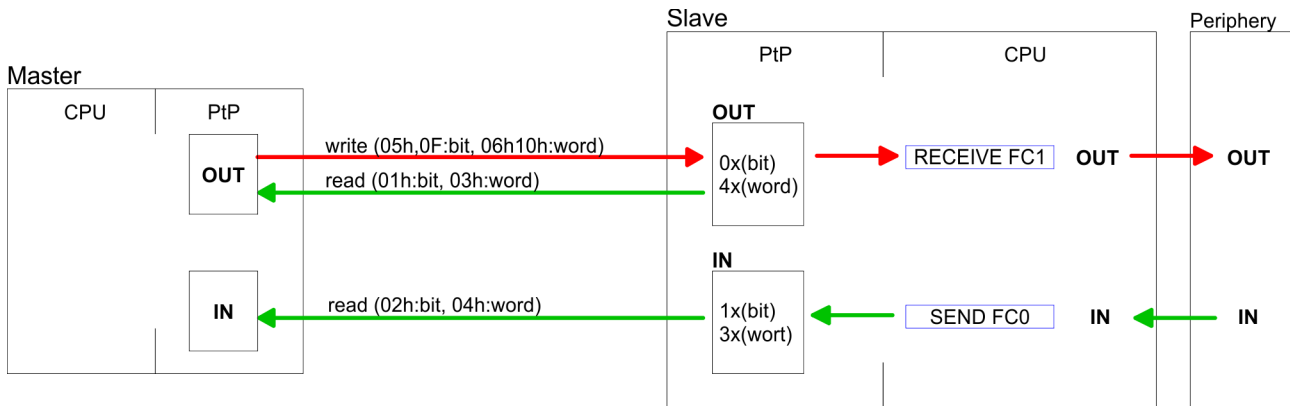
Overview

With the following Modbus function codes a Modbus master can access a Modbus slave: With the following Modbus function codes a Modbus master can access a Modbus slave. The description always takes place from the point of view of the master:

Code	Command	Description
01h	Read n bits	Read n bits of master output area 0x
02h	Read n bits	Read n bits of master input area 1x
03h	Read n words	Read n words of master output area 4x
04h	Read n words	Read n words master input area 3x
05h	Write 1 bit	Write 1 bit to master output area 0x
06h	Write 1 word	Write 1 word to master output area 4x
0Fh	Write n bits	Write n bits to master output area 0x
10h	Write n words	Write n words to master output area 4x

Point of View of "Input" and "Output" data

The description always takes place from the point of view of the master. Here data, which were sent from master to slave, up to their target are designated as "output" data (OUT) and contrary slave data received by the master were designated as "input" data (IN).



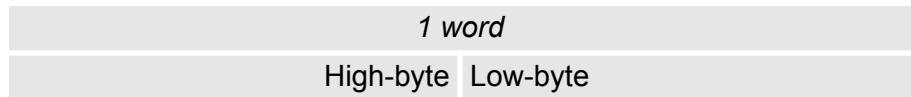
Respond of the slave

If the slave announces an error, the function code is send back with an "ORed" 80h.

Without an error, the function code is sent back.

Slave answer:	Function code OR 80h	→ Error
	Function code	→ OK

Byte sequence in a word



Check sum CRC, RTU, LRC

The shown check sums CRC at RTU and LRC at ASCII mode are automatically added to every telegram. They are not shown in the data block.

Read n bits 01h, 02h

Code 01h: Read n bits of master output area 0x
Code 02h: Read n bits of master input area 1x

Command telegram

Slave address	Function code	Address 1. bit	Number of bits	Check sum CRC/LRC
1byte	1byte	1word	1word	1word

Respond telegram

Slave address	Function code	Number of read bytes	Data 1. byte	Data 2. byte	...	Check sum CRC/LRC
1byte	1byte	1byte	1byte	1byte		1word
			max. 250byte			

Read n words 03h, 04h 03h: Read n words of master output area 4x
 04h: Read n words master input area 3x

Command telegram

Slave address	Function code	Address 1. bit	Number of words	Check sum CRC/LRC
1byte	1byte	1word	1word	1word

Respond telegram

Slave address	Function code	Number of read bytes	Data 1. word	Data 2. word	...	Check sum CRC/LRC
1byte	1byte	1byte	1word	1word		1word
			max. 125words			

Write 1 bit 05h Code 05h: Write 1 bit to master output area 0x
 A status change is via "Status bit" with following values:
 "Status bit" = 0000h → Bit = 0
 "Status bit" = FF00h → Bit = 1

Command telegram

Slave address	Function code	Address bit	Status bit	Check sum CRC/LRC
1byte	1byte	1word	1word	1word

Respond telegram

Slave address	Function code	Address bit	Status bit	Check sum CRC/LRC
1byte	1byte	1word	1word	1word

Write 1 word 06h

Code 06h: Write 1 word to master output area 4x

Command telegram

Slave address	Function code	Address word	Value word	Check sum CRC/LRC
1byte	1byte	1word	1word	1word

Respond telegram

Slave address	Function code	Address word	Value word	Check sum CRC/LRC
1byte	1byte	1word	1word	1word

Write n bits 0Fh

Code 0Fh: Write n bits to master output area 0x

Please regard that the number of bits has additionally to be set in byte.

Command telegram

Slave address	Function code	Address 1. bit	Number of bits	Number of bytes	Data 1. byte	Data 2. byte	...	Check sum CRC/LRC
1byte	1byte	1word	1word	1byte	1byte	1byte	1byte	1word
					max. 250byte			

Respond telegram

Slave address	Function code	Address 1. bit	Number of bits	Check sum CRC/LRC
1byte	1byte	1word	1word	1word

Write n words 10h

Code 10h: Write n words to master output area 4x

Command telegram

Slave address	Function code	Address 1. word	Number of words	Number of bytes	Data 1. word	Data 2. word	...	Check sum CRC/LRC
1byte	1byte	1word	1word	1byte	1word	1word	1word	1word
					max. 125words			

Respond telegram

Slave address	Function code	Address 1. word	Number of words	Check sum CRC/LRC
1byte	1byte	1word	1word	1word

7.8 Modbus - Example communication

Overview

The example establishes a communication between a master and a slave via Modbus. The following combination options are shown:

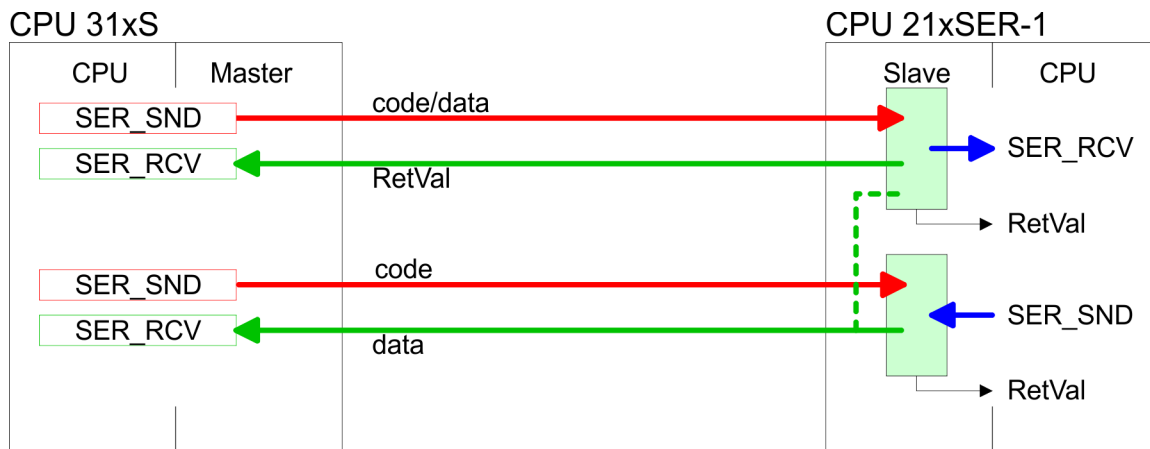
- CPU 31xS as Modbus RTU master
- CPU 21xSER-1 as Modbus RTU slave
- Siemens SIMATIC Manager and possibilities for the project transfer
- Modbus cable connection

Approach

1. ▶ Assemble a Modbus system consisting of a CPU 31xS as Modbus master and a CPU 21xSER-1 as Modbus slave and Modbus cable.
2. ▶ Execute the project engineering of the master! For this you create a PLC user application with the following structure:
 - OB 100:
Call SFC 216 (configuration as Modbus RTU master) with timeout setting and error evaluation.
 - OB 1:
Call SFC 217 (SER_SND) where the data is send with error evaluation. Here you have to build up the telegram according to the Modbus rules. Call SFC 218 (SER_RECV) where the data is received with error evaluation.

3. ▶ Execute the project engineering of the slave! The PLC user application at the slave has the following structure:
 - OB 100:
Call SFC 216 (configuration as Modbus RTU slave) with timeout setting and Modbus address in the DB and error evaluation.
 - OB 1:
Call SFC 217 (SER_SND) for data transport from the slave CPU to the output buffer. Call SFC 218 (SER_RECV) for the data transport from the input buffer to the CPU. Allow an according error evaluation for both directions.

Structure for the according PLC programs for master and slave:





8 WinPLC7

8.1 System conception

General

WinPLC7 is a programming and simulation software from VIPA for every PLC programmable with Siemens STEP®7. This tool allows you to create user applications in FBD, LAD and STL. Besides of a comfortable programming environment, WinPLC7 has an integrated simulator that enables the simulation of your user application at the PC without additional hardware. This "Soft-PLC" is handled like a real PLC and offers the same error behavior and diagnostics options via diagnostics buffer, USTACK and BSTACK.



Detailed information and programming samples may be found at the online help respectively in the online documentation of WinPLC7.

Alternatives

There is also the possibility to use according configuration tools from Siemens instead of WinPLC7 from VIPA. Here the proceeding is part of this manual.

System requirements

- Windows XP (SP3)
- Windows Vista
- Windows 7 (32 and 64 bit)
- Windows 8 (32 and 64 bit)

Source

You may receive a *demo version* from VIPA. Without any activation with the *demo version* the CPUs 11x of the System 100V from VIPA may be configured. To configure the SPEED7 CPUs a license for the "profi" version is necessary. This may be online be received from VIPA and activated.

There are the following sources to get WinPLC7:

- Online
 - At www.vipa.com in the service area at Downloads a link to the current demo version and the updates of WinPLC7 may be found.
- CD
 - SW211C1DD: WinPLC7 Single license, CD, with documentation in german
 - SW211C1ED: WinPLC7 Single license, CD, with documentation in english

8.2 Installation

Precondition

The project engineering of a SPEED7 CPU from VIPA with WinPLC7 is only possible using an activated "Profi" version of WinPLC7.

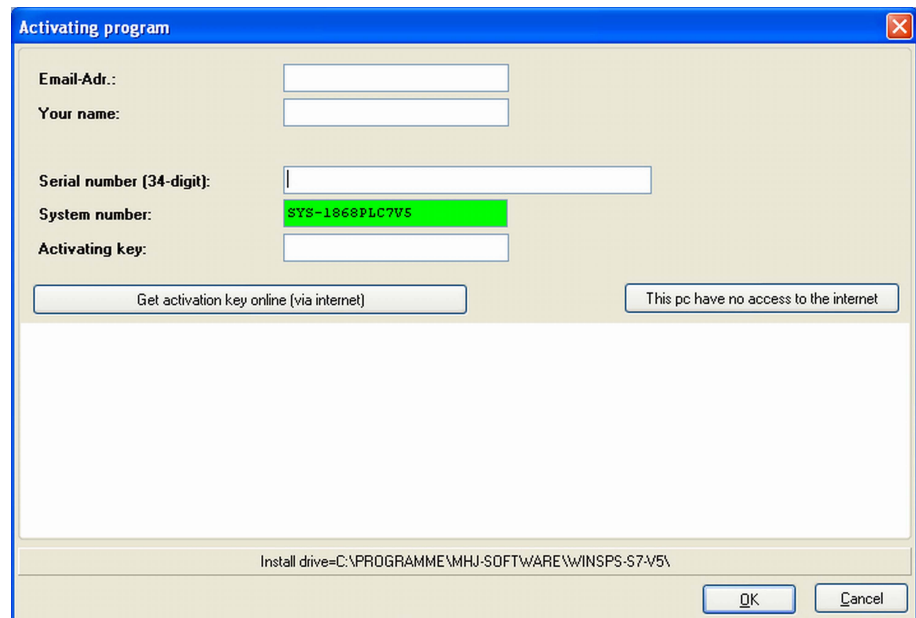
Installation WinPLC7 Demo

The installation and the registration of WinPLC7 has the following approach:

1. ▶ For installation of WinPLC7 start the setup program of the corresponding CD respectively execute the online received exe file.
2. ▶ Select the according language.
3. ▶ Accept the licensing agreement.
4. ▶ Set an installation directory and a group assignment and start the installation.

Activation of the "Profi" version

1. ▶ Start WinPLC7.
⇒ A 'Demo' dialog is shown
2. ▶ Click at [Activate Software].
⇒ The following dialog for activation is shown:



3. ▶ Fill in the following fields:
 - Email-Addr.
 - Your Name
 - Serial number
The serial number may be found on a label at the CD case of WinPLC7.
4. ▶ If your computer is connected to Internet you may online request the Activation Key by [Get activation key via Internet]. Otherwise click at [This PC has no access to the Internet] and follow the instructions.
 - ⇒ With successful registration the activation key is listed in the dialog window respectively is sent by email.
5. ▶ Enter this at 'Activation code' and click at [OK].
 - ⇒ Now, WinPLC7 is activated as "Profi" version.

Installation of WinPCAP for station search via Ethernet

To find a station via Ethernet (accessible nodes) you have to install the WinPCAP driver. This driver may be found on your PC in the installation directory at WinSPS-S7-V5/WinPcap_... .exe. Execute this file and follow the instructions.

8.3 Example project engineering

8.3.1 Job definition

In the example a FC 1 is programmed, which is cyclically called by the OB 1. By setting of 2 comparison values (value1 and value2) during the FC call, an output of the PLC-System should be activated depending on the comparison result.

Here it should apply:

- if value1 = value2 activate output Q 124.0
- if value1 > value2 activate output Q 124.1
- if value1 < value2 activate output Q 124.2

Precondition

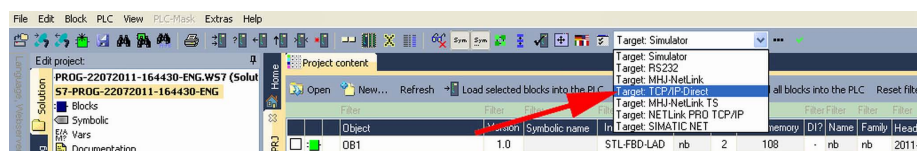
- You have administrator rights for your PC.
- WinPLC7 is installed and activated as "Profi" version.
- One SPEED7 CPU and one digital output module are installed and cabled.
- The Ethernet PG/OP channel of the CPU is connected to your Ethernet network. Your CPU may be connected to your PC with an Ethernet cable either directly or via hub/switch.
- WinPCap for station search via Ethernet is installed.
- The power supply of the CPU and the I/O periphery are activated and the CPU is in STOP state.

8.3.2 Project engineering

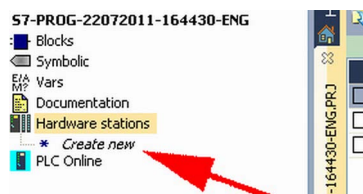
1. ▶ Start WinPLC7 ("Profi" version)
2. ▶ Create and open a new project with [Create a new solution].

Hardware configuration

1. ▶ For the call of the hardware configurator it is necessary to set WinPLC7 from the Simulator-Mode to the Offline-Mode. For this and the communication via Ethernet set "Target: TCP/IP Direct".



2. ▶ Double click to 'Hardware stations' and here at 'Create new'.



3. ▶ Enter a station name. Please consider that the name does not contain any spaces.

4. ▶ After the load animation choose in the register Select PLC-System the system "VIPA SPEED7" and click to [Create]. A new station is created.
5. ▶ Save the empty station with [Strg]+[S].
6. ▶ By double click or drag&drop the according VIPA CPU in the hardware catalog at 'CPU SPEED7' the CPU is inserted to your configuration.
7. ▶ For output place a digital output module, assign the start address 124 and save the hardware configuration.

Establish online access via Ethernet PG/OP channel:

1. ▶ Open the CPU-Properties, by double clicking to the CPU at slot 2 in the hardware configurator.
2. ▶ Click to the button [Ethernet CP-Properties (PG/OP-channel)].
⇒ The dialog 'Properties CP343' is opened.
3. ▶ Chose the register 'Common Options'.
4. ▶ Click to [Properties Ethernet].
5. ▶ Choose the subnet 'PG_OP_Ethernet'.
6. ▶ Enter a valid IP address-and a subnet mask. You may get this from your system administrator.
7. ▶ Close every dialog window with [OK].
8. ▶ Select, if not already done, 'Target: External TCP/IP direct'.
9. ▶ Open with 'Online → Send configuration to the CPU' a dialog with the same name.
10. ▶ Click to [Accessible nodes]. Please regard to use this function it is necessary to install WinPCap before!
11. ▶ Choose your network card and click to [Determining accessible nodes].
⇒ After a waiting time every accessible station is listed. Here your CPU with IP 0.0.0.0 is listed, too. To check this the according MAC address is also listed. This MAC address may be found at a label beneath the front flap of the CPU.
12. ▶ For the temporary setting of an IP address select you CPU and click to [Temporary setting of the IP parameters]. Please enter the same IP parameters, you configured in the CPU properties and click to [Write Parameters].
13. ▶ Confirm the message concerning the overall reset of the CPU.
⇒ The IP parameters are transferred to the CPU and the list of accessible stations is refreshed.
14. ▶ Select your CPU and click to [Confirm].
⇒ Now you are back in the dialog "Send configuration".

Transfer hardware configuration

- ▶ Choose your network card and click to [Send configuration].
⇒ After a short time a message is displayed concerning the transfer of the configuration is finished.

The hardware configuration is finished, now and the CPU may always be accessed by the IP parameters as well by means of WinPLC7.

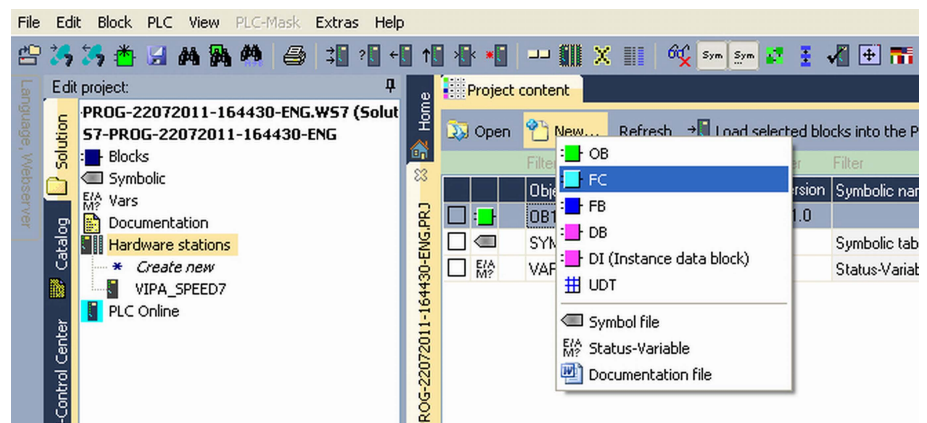


Usually the online transfer of the hardware configuration happens within the hardware configurator. With 'File → Save active station in the WinPL7 sub project' there is also the possibility to store the hardware configuration as a system file in WinPLC7 to transfer it from WinPLC7 to the CPU.

Programming of the FC 1

The PLC programming happens by WinPLC7. Close the hardware configurator and return to your project in WinPLC7. The PLC program is to be created in the FC 1.

1. In 'Project content' choose 'New → FC'.



2. Enter "FC1" as block and confirm with [OK].
⇒ The editor for FC 1 is called.

Creating parameters

In the upper part of the editor there is the parameter table. In this example the 2 integer values *value1* and *value2* are to be compared together. Since both values are read only by the function, these are to be defined as "in".

1. Select the 'in →' row at the 'parameter table' and enter at the field 'Name' "value1". Press the [Return] key.
⇒ The cursor jumps to the column with the data type.
2. The data type may either directly be entered or be selected from a list of available data types by pressing the [Return] key. Set the data type to INT and press the [Return] key.
⇒ Now the cursor jumps to the 'Comment' column.
3. Here enter "1. compare value" and press the [Return] key.
⇒ A new 'in →' row is created and the cursor jumps to 'Name'.
4. Proceed for *value2* in the same way as described for *value1*.

- Save the block. A note that the interface of the block was changed may be acknowledged with [Yes].

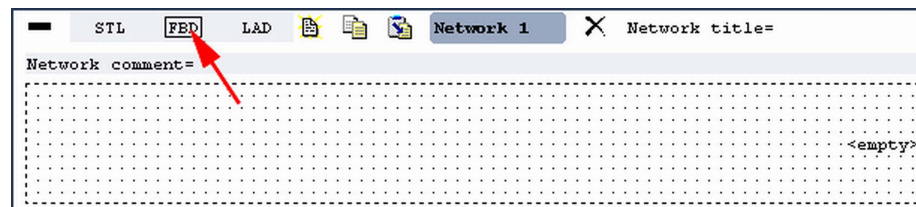
⇒ The parameter table shows the following entries, now:

Address	Declaration	Name	Type	Initial value	Comment
0.0	in ->	value1	INT		1. compare value
2.0	in ->	value2	INT		2. compare value
	out <->				
	in_out <->				

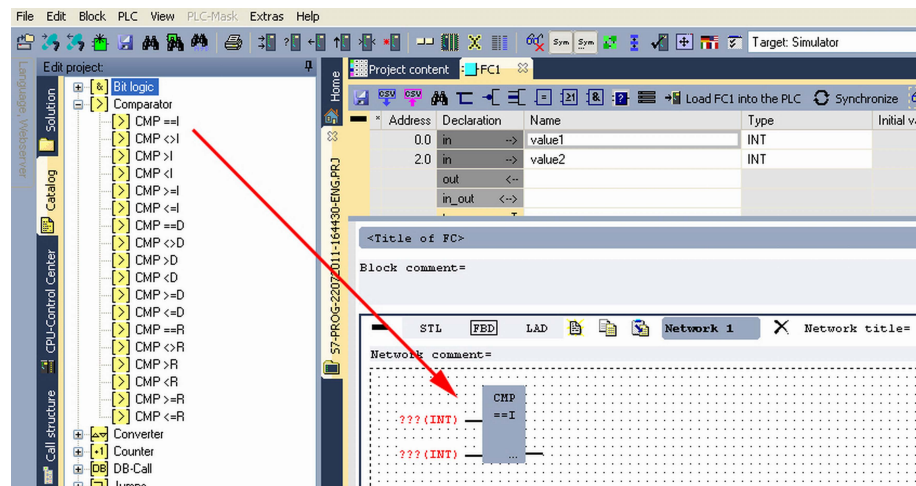
Enter the program

As requested in the job definition, the corresponding output is activated depending on the comparison of *value1* and *value2*. For each comparison operation a separate network is to be created.

- The program is to be created as FBD (function block diagram). Here change to the FBD view by clicking at 'FBD'.



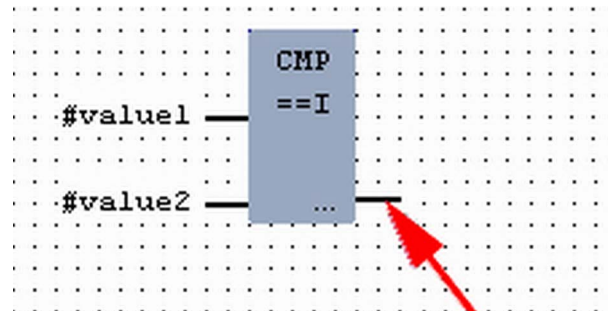
- Click to the input field designated as "<empty>". The available operations may be added to your project by drag&drop from the *hardware catalog* or by double click at them in the *hardware catalog*.
- Open in the *catalog* the category "Comparator" and add the operation 'CMP==I' to your network.



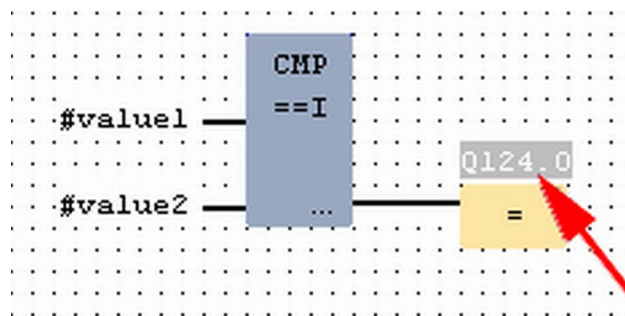
- Click to the input left above and insert *value1*. Since these are block parameters a selection list of block parameters may be viewed by entering "#".
- Type in "#" and press the [Return] key.
- Choose the corresponding parameter of the list and confirm it with the [Return] key.
- Proceed in the same way with the parameter *value2*.

The allocation to the corresponding output, here Q 124.0, takes place with the following proceeding:

1. Click to the output at the right side of the operator.



2. Open in the *catalog* the category 'Bit logic' and select the function '--[=]'. The inserting of '--[=]' corresponds to the WinPLC7 shortcut [F7].
3. Insert the output Q 124.0 by clicking to the operand.



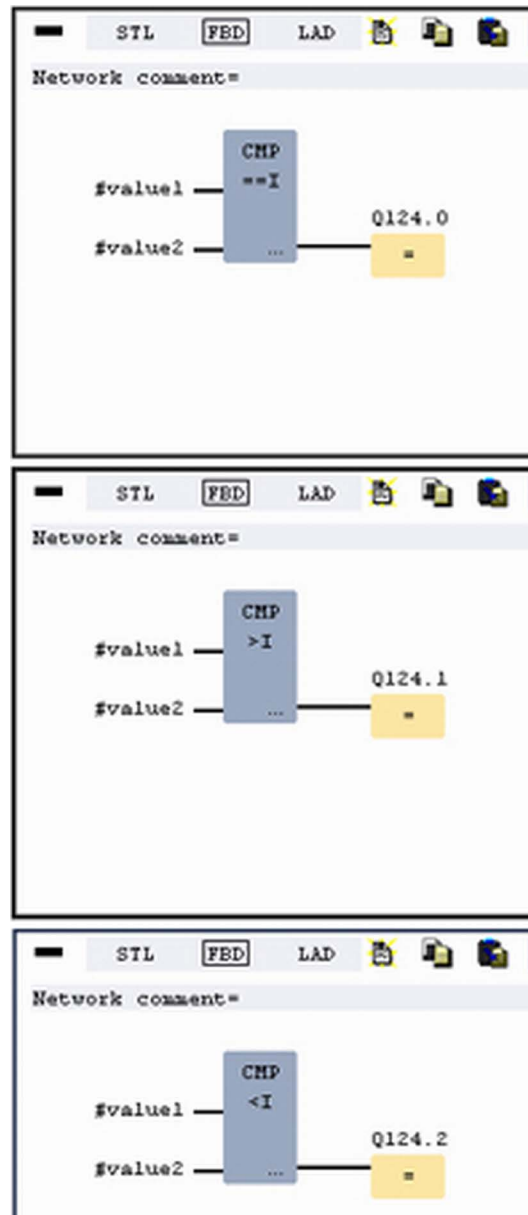
⇒ Network1 is finished, now.

Adding a new network

For further comparisons the operations "CMP>I" at Q 124.1 and "CMP<I" at Q 124.2 are necessary. Create a network for both operations with the following proceeding:

1. Move your mouse at an arbitrary position on the editor window and press the right mouse key.
2. Select at 'context menu → Insert new network'.
 - ⇒ A dialog field is opened to enter the position and number of the networks.
3. Proceed as described for "Network 1".

4. ▶ Save the FC 1 with 'File → Save content of focused window' respectively press [Strg]+[S].
- ⇒ After you have programmed the still missing networks, the FC 1 has the following structure:

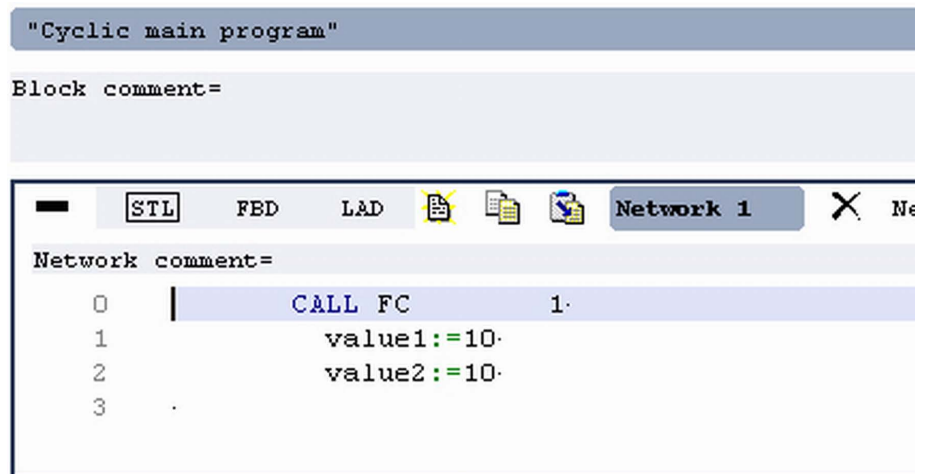


Creating the block OB 1 The FC 1 is to be called from the cycle OB 1.

1. ▶ Go to OB 1, which was automatically created with starting the project.
2. ▶ Go to 'Project content' or to 'Solution' and open the OB 1 by a double click.
3. ▶ Change to the STL view.

Example project engineering > Test the PLC program in the Simulator

4. ▶ Type in "Call FC 1" and press the *[Return]* key.
 - ⇒ The FC parameters are automatically displayed and the following parameters are assigned:



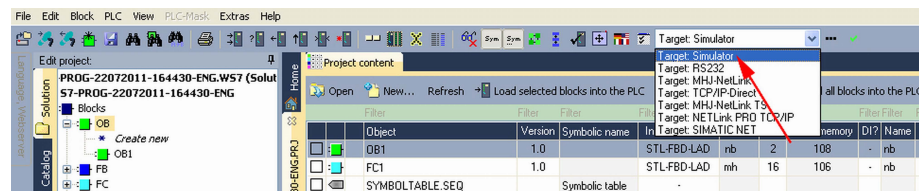
5. ▶ Save the OB 1 with respectively press *[Strg]+[S]*.

8.3.3 Test the PLC program in the *Simulator*

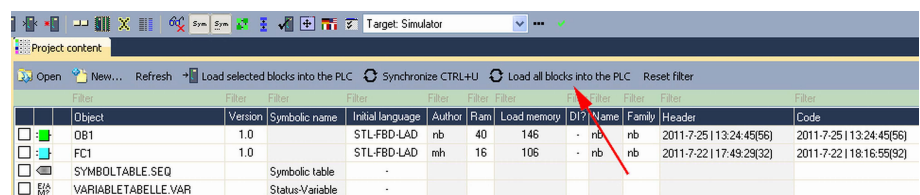
Proceeding

With WinPLC7 there is the possibility to test your project in a *Simulator*.

1. ▶ Here select 'Target: Simulator'.



2. ▶ Transfer the blocks to the simulator with [Load all blocks into the PLC].



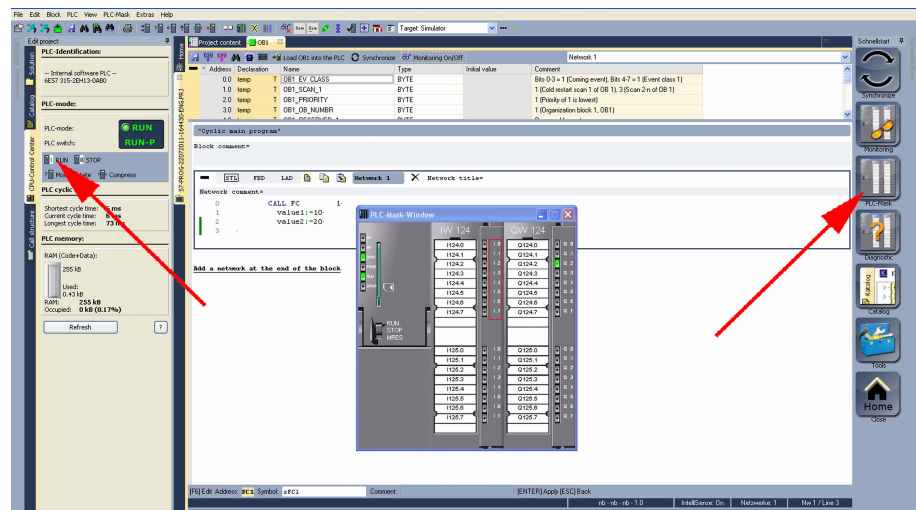
3. ▶ Switch the CPU to RUN, by clicking at 'RUN' in the 'CPU Control Center' of 'Edit project'.
 - ⇒ The displayed state changes from STOP to RUN.
4. ▶ To view the process image select 'View → Display process image window' or click at .
 - ⇒ The various areas are displayed.
5. ▶ Double click to the process image and enter at 'Line 2' the address PQB 124. Confirm your input with [OK]. A value marked by red color corresponds to a logical "1".
6. ▶ Open the OB 1.

7. ➤ Change the value of one variable, save the OB 1 and transfer it to the simulator.
 - ⇒ According to your settings the process image changes immediately. The status of your blocks may be displayed with 'Block → Monitoring On/Off'.

Visualization via PLC mask

A further component of the simulator is the *PLC mask*. Here a CPU is graphically displayed, which may be expanded by digital and analog peripheral modules. As soon as the CPU of the simulator is switched to RUN state, inputs may be activated by mouse and outputs may be displayed.

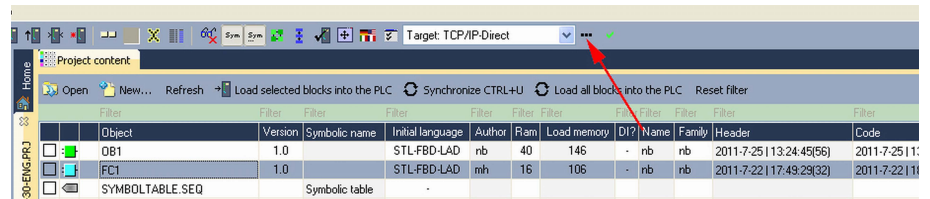
1. ➤ Open the *PLC mask* with 'view → PLC mask'.
 - ⇒ A CPU is graphically displayed.
2. ➤ Double-click to the output module, open its properties dialog and enter the Module address 124.
3. ➤ Switch the operating mode switch to RUN by means of the mouse.
 - ⇒ Your program is executed and displayed in the simulator, now.



8.3.4 Transfer PLC program to CPU and its execution

Proceeding

1. ➤ For transfer to the CPU set the transfer mode to "Target: TCP/IP-Direct".
2. ➤ If there are more network adapters in your PC, the network adapter may be selected via 'Extras → Select network adapter'.
3. ➤ For presetting the Ethernet data click to [...] and click to [Accessible nodes].



4. Click at [Determining accessible nodes].
⇒ After a waiting time every accessible station is listed.
5. Choose your CPU, which was provided with TCP/IP address parameters during the hardware configuration and click to [Confirm].
6. Close the dialog 'Ethernet properties' with [OK].
7. Transfer your project to your CPU with 'PLC → Send all blocks'.
8. Switch your CPU to RUN state.
9. Open the OB 1 by double click.
10. Change the value of one variable, save the OB 1 and transfer it to the CPU.
⇒ According to your settings the process image changes immediately. The status of your blocks may be displayed with 'Block → Monitoring On/Off'.

9 Configuration with TIA Portal

9.1 TIA Portal - Work environment

9.1.1 General

General

In this chapter the project engineering of the VIPA CPU in the Siemens TIA Portal is shown. Here only the basic usage of the Siemens TIA Portal together with a VIPA CPU is shown. TIA means **T**otally **I**ntegrated **A**utomation from Siemens. Here your VIPA PLCs may be configured and linked. For diagnostics online tools are available.

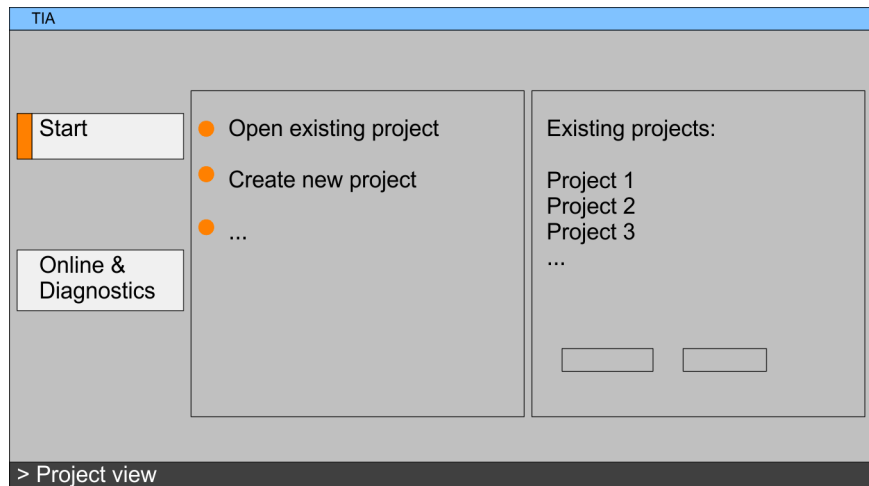


Information about the Siemens TIA Portal can be found in the online help respectively in the according online documentation.

Starting the TIA Portal

To start the Siemens TIA Portal with Windows select 'Start → Programs → Siemens Automation → TIA ...'

Then the TIA Portal opens with the last settings used.



Exiting the TIA Portal

With the menu 'Project → Exit' in the 'Project view' you may exit the TIA Portal. Here there is the possibility to save changes of your project before.

9.1.2 Work environment of the TIA Portal

Basically, the TIA Portal has the following 2 views. With the button on the left below you can switch between these views:

Portal view

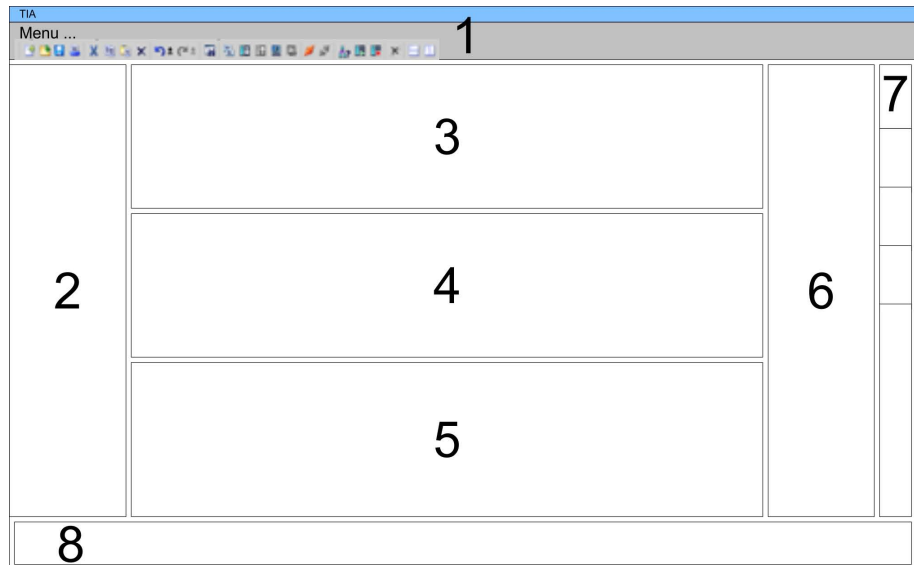
The 'Portal view' provides a "task oriented" view of the tools for processing your project. Here you have direct access to the tools for a task. If necessary, a change to the Project view takes place automatically for the selected task.

Project view

The 'Project view' is a "structured" view to all constituent parts of your project.

Areas of the Project view

The Project view is divided into the following areas:



- 1 Menu bar with toolbars
- 2 Project tree with Details view
- 3 Project area
- 4 Device overview of the project respectively area for block programming
- 5 Properties dialog of a device (parameter) respectively information area
- 6 Hardware catalog and tools
- 7 "Task-Cards" to select hardware catalog, tasks and libraries
- 8 Jump to Portal or Project view

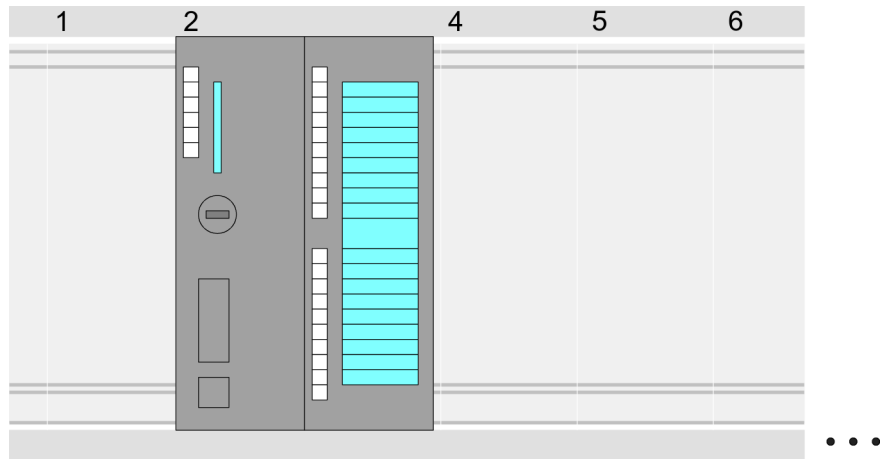
9.2 TIA Portal - Hardware configuration - CPU

Configuration Siemens CPU

With the Siemens TIA Portal, the CPU 312-5BE13 from VIPA is to be configured as CPU 312C (6ES7 312-5BE03-0AB0 V2.6) from Siemens.

- 1. ▶ Start the Siemens TIA Portal.
- 2. ▶ Create a new project in the *Portal view* with 'Create new project'.
- 3. ▶ Switch to the *Project view*.
- 4. ▶ Click in the *Project tree* at 'Add new device'.
- 5. ▶ Select the following CPU in the input dialog:
 SIMATIC S7-300 > CPU 312C > 6ES7 312-5BE03-0AB0 V2.6
 ⇒ The CPU is inserted with a profile rail.

Project area:



Device overview:

Module	...	Slot	...	Type	...
PLC ...		2		CPU 312C	
MPI inter- face...		2 0		MPI interface	
DI10/DO6...		2 X2		DI10/DO6	
Counter...		2 4		Counter	
...					

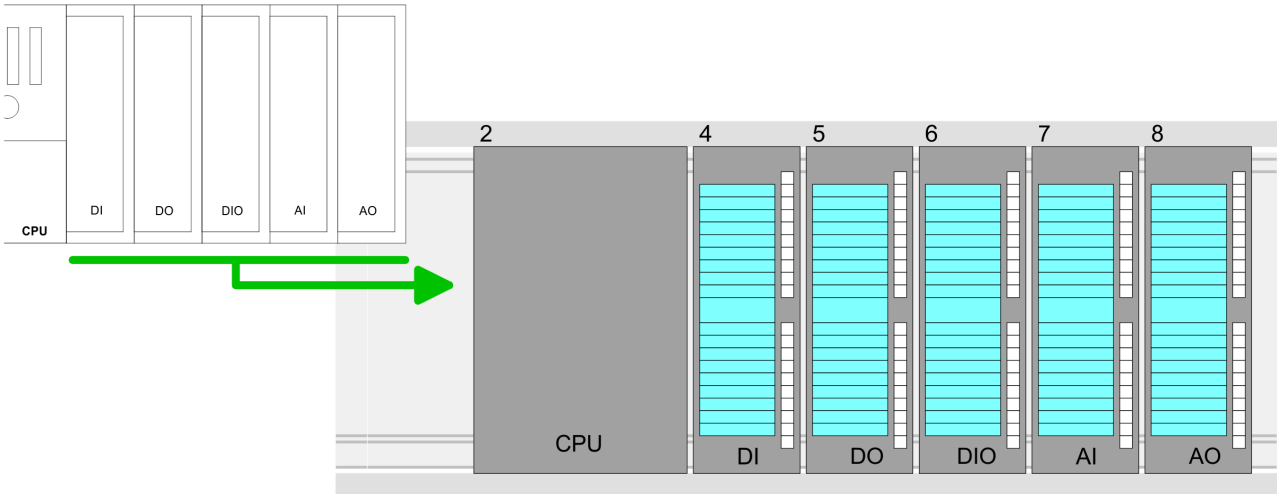
Setting standard CPU parameters

Since the CPU 312-5BE13 from VIPA is configured as Siemens CPU 312C, so the parametrization takes place via the Siemens CPU. For parametrization click in the *Project area* respectively in the *Device overview* at the CPU part. Then the parameters of the CPU part are shown in the *Properties dialog*. Here you can make your parameter settings. ↪ [Chapter 5.8 'CPU parametrization' on page 52](#)

9.3 TIA Portal - Hardware configuration - I/O modules

Hardware configuration of the modules

After the hardware configuration of the CPU place the System 300 modules at the bus in the plugged sequence. For this drag&drop the according module from the Hardware catalog to the according position of the profile rail in the *Project area* or in the *Device overview*



Device overview

Module	...	Slot	...	Type	...
PLC...		2		CPU ...	
...		
		3			
DI...		4		DI...	
DO...		5		DO...	
DIO...		6		DIO...	
AI...		7		AI...	
AO...		8		AO...	

Parametrization

For parametrization click in the *Project area* respectively in the *Device overview* on the module you want to parameterize. The parameters of the module appear in the Properties dialog. Here you can make your parameter settings.

9.4 TIA Portal - Hardware configuration - Ethernet PG/OP channel

Overview

The CPU has an integrated Ethernet PG/OP channel. This channel allows you to program and remote control your CPU.

- The Ethernet PG/OP channel also gives you access to the internal web page that contains information about firmware version, connected I/O devices, current cycle times etc.
- At the first commissioning respectively after a factory reset the Ethernet PG/OP channel has no IP address.
- For online access to the CPU via the Ethernet PG/OP channel, valid IP address parameters have to be assigned to this. This is called "initialization".
- This can be done with the Siemens TIA Portal.

Assembly and commissioning

1. ▶ Install your System 300S with your CPU.
2. ▶ Wire the system by connecting cables for voltage supply and signals.
3. ▶ Connect the Ethernet jack of the Ethernet PG/OP channel to Ethernet.
4. ▶ Switch on the power supply.
 - ⇒ After a short boot time the CP is ready for communication. He possibly has no IP address data and requires an initialization.

"Initialization" via Online functions

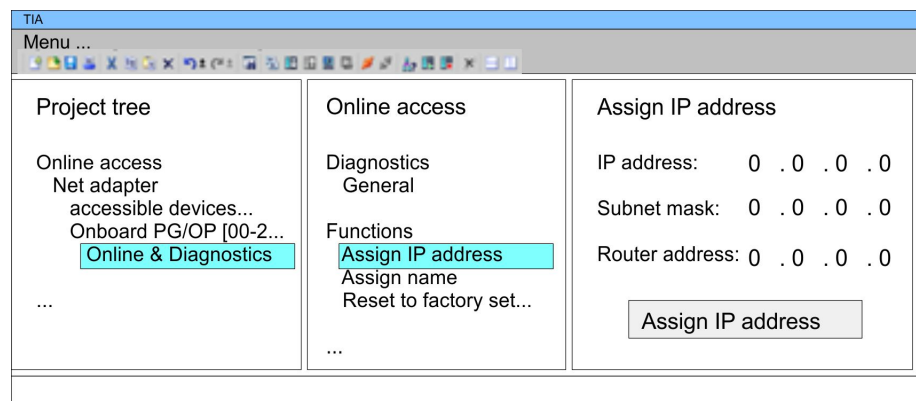
The initialization via the Online functions takes place with the following proceeding:

- ▶ Determine the current Ethernet (MAC) address of your Ethernet PG/OP channel. This can be found as 1. address under the front flap of the CPU on a sticker on the left side.

Assign IP address parameters

You get valid IP address parameters from your system administrator. The assignment of the IP address data happens online in the Siemens TIA Portal with the following proceeding:

1. ▶ Start the Siemens TIA Portal.
2. ▶ Switch to the 'Project view'.
3. ▶ Click in the 'Project tree' at 'Online access' and choose here by a doubleclick your network card, which is connected to the Ethernet PG/OP channel.
4. ▶ To get the stations and their MAC address, use the 'Accessible device'. The MAC address can be found at the 1. label beneath the front flap of the CPU.
5. ▶ Choose from the list the module with the known MAC address (Onboard PG/OP [MAC address]) and open with "Online & Diagnostics" the diagnostics dialog in the Project area.
6. ▶ Navigate to *Functions > Assign IP address*. Type in the IP configuration like IP address, subnet mask and gateway.
7. ▶ Confirm with [Assign IP configuration].
 - ⇒ Directly after the assignment the Ethernet PG/OP channel is online reachable using the set IP address data. The value remains as long as it is reassigned, it is overwritten by a hardware configuration or an factory reset is executed.

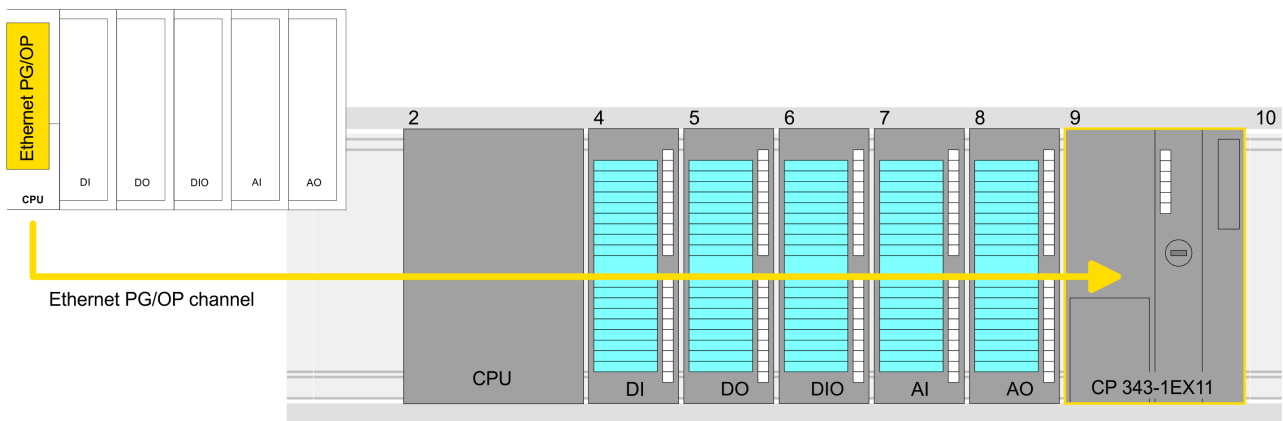




Due to the system you may get a message that the IP address could not be assigned. This message can be ignored.

Take IP address parameters in project

1. ➤ Open your project.
2. ➤ If not already done, configure in the 'Device configuration' a Siemens CPU 312C (6ES7 312-5BE03-0AB0 V2.6).
3. ➤ Configure the System 300 modules.
4. ➤ For the Ethernet PG/OP channel you have to configure a Siemens CP 343-1 (6GK7 343-1EX11 0XE0) always as last module after the really plugged modules.
5. ➤ Open the "Property" dialog by clicking on the CP 343-1EX11 and enter for the CP at "Properties" at "Ethernet address" the IP address data, which you have assigned before.
6. ➤ Transfer your project.



Device overview:

Module	...	Slot	...	Type	...
PLC...		2		CPU ...	
...		
		3			
DI...		4		DI...	
DO...		5		DO...	
DIO...		6		DIO...	
AI...		7		AI...	
AO...		8		AO...	
■ CP 343-1		9		CP 343-1	

9.5 TIA Portal - Include VIPA library

Overview

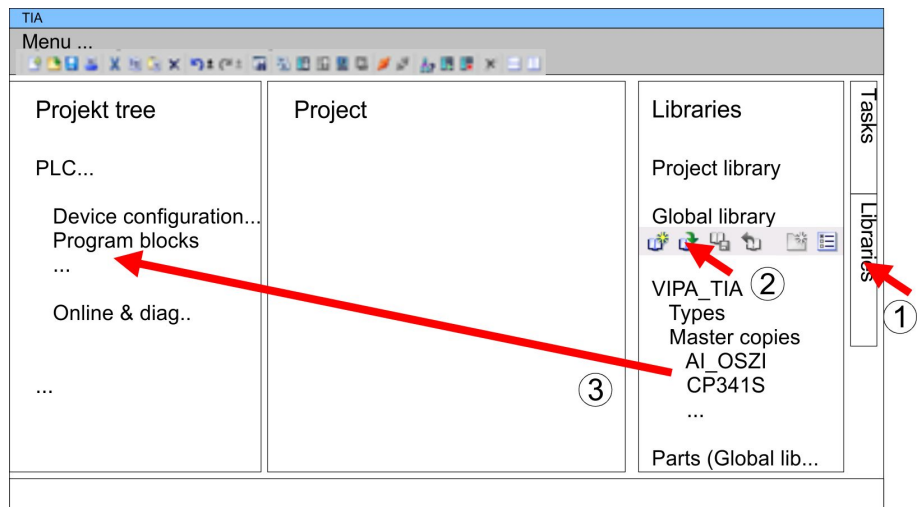
- The VIPA specific blocks may be found at www.vipa.com as downloadable library at the "service" area with *Downloads > VIPA LIB*.
- The library is available as packed zip-file *Fx000020_V...*
- If you want to use VIPA specific blocks, you have to import the library into your project.
Execute the following steps:
 - Extract *Fx000020_V... .zip*
 - Open library and transfer blocks into the project

**Unzip
Fx000020_V... .zip**

Start your un-zip application with a double click on the file *Fx000020_V... .zip* and copy all the files and folders in a work directory for the Siemens TIA Portal.

Open library and transfer blocks to project

1. ▶ Start the Siemens TIA Portal with your project.
2. ▶ Select the *Project view*.
3. ▶ Choose "Libraries" from the Task cards on the right side.
4. ▶ Click at "Global libraries".
5. ▶ Click at "Open global library".
6. ▶ Navigate to your directory and load the file *VIPA_TIA.al11*.



7. ▶ Copy the necessary blocks from the library into the "Program blocks" of the Project tree of your project. Now you have access to the VIPA specific blocks via your user application.

9.6 TIA Portal - Project transfer

Overview

There are the following possibilities for project transfer into the CPU:

- Transfer via MPI
- Transfer via Ethernet
- Transfer via memory card

Transfer via MPI

Currently the VIP A programming cables for transfer via MPI are not supported. This is only possible with the programming cable from Siemens.

1. ▶ Establish a connection to the CPU via MPI with an appropriate programming cable. Information may be found in the corresponding documentation of the programming cable.
2. ▶ Switch-ON the power supply of your CPU and start the Siemens TIA Portal with your project.
3. ▶ Select in the *Project tree* your CPU and choose '*Context menu → Download to device → Hardware configuration*' to transfer the hardware configuration.
4. ▶ To transfer the PLC program choose '*Context menu → Download to device → Software*'. Due to the system you have to transfer hardware configuration and PLC program separately.

Transfer via Ethernet

For transfer via Ethernet the CPU has the following interface:

- X5: Ethernet PG/OP channel

Initialization

So that you may the according Ethernet interface, you have to assign IP address parameters by means of the "initialization".

Please consider to use the same IP address data in your project for the CP 343-1.

Transfer

1. ▶ For the transfer, connect, if not already done, the appropriate Ethernet jack to your Ethernet.
2. ▶ Open your project with the Siemens TIA Portal.
3. ▶ Click in the *Project tree* at *Online access* and choose here by a double-click your network card, which is connected to the Ethernet PG/OP interface.
4. ▶ Select in the *Project tree* your CPU and click at [Go online].
5. ▶ Set the access path by selecting "PN/IE" as type of interface, your network card and the according subnet. Then a net scan is established and the corresponding station is listed.
6. ▶ Establish with [Connect] a connection.
7. ▶ Click to '*Online → Download to device*'.
 - ⇒ The according block is compiled and by a request transferred to the target device. Provided that no new hardware configuration is transferred to the CPU, the entered Ethernet connection is permanently stored in the project as transfer channel.

Transfer via memory card

The memory card serves as external storage medium. There may be stored several projects and sub-directories on a memory card. Please regard that your current project is stored in the root directory and has one of the following file names:

- S7PROG.WLD
- AUTOLOAD.WLD

1. ▶ Create in the Siemens TIA Portal a wld file with '*Project* → *Memory card file* → *New*'.

⇒ The wld file is shown in the *Project tree* at "SIMATIC Card Reader" as "Memory card file".

2. ▶ Copy the blocks from the *Program blocks* to the wld file. Here the hardware configuration data are automatically copied to the wld file as "System data".

Transfer memory card → CPU

The transfer of the application program from the memory card into the CPU takes place depending on the file name after an overall reset or PowerON.

- *S7PROG.WLD* is read from the memory card after overall reset.
- *AUTOLOAD.WLD* is read from the memory card after PowerON.

The blinking of the MC LED of the CPU marks the active transfer. Please regard that your user memory serves for enough space for your user program, otherwise your user program is not completely loaded and the SF LED gets on.

Transfer CPU → Memory card

When a memory card has been installed, the write command stores the content of the RAM as *S7PROG.WLD* on the memory card. The write command can be found in the Siemens TIA Portal in the Task card "Online tools" in the command area at "Memory" as button [Copy RAM to ROM]. The MC LED blinks during the write access. When the LED expires, the write process is finished. If this project is to be loaded automatically from the memory card with PowerON, you have to rename this to on the memory card to *AUTOLOAD.WLD*.

Checking the transfer operation

After accessing the memory card you can find a diagnostics entry in the CPU. To monitor the diagnostics entries, you select *Online & Diagnostics* in the Siemens TIA Portal. Here you can access the "Diagnostics buffer". ↪ *Chapter 5.19 'VIPA specific diagnostic entries' on page 72*